EXHIBIT 20

IN THE UNITED STATES DISTRICT COURT
FOR THE WESTERN DISTRICT OF TEXAS
WACO DIVISION

PROFESSOR MASAHIRO IIDA, Plaintiff, v. INTEL CORPORATION, Defendant.	:	Case No. 6:22-cv-00662 JURY TRIAL DEMANDED

PLAINTIFF'S DISCLOSURE OF ASSERTED CLAIMS AND PRELIMINARY INFRINGEMENT CONTENTIONS

Pursuant to this Court's Standing Order Governing Proceedings (OGP) 4.2 – Patent Cases, Plaintiff, Professor Masahiro Iida ("Professor Iida"), hereby provides his preliminary infringement contentions regarding U.S. Patent No. 6,812,737 (the "'737 patent") to Defendant, Intel Corporation ("Intel").

These infringement contentions are preliminary and based on Professor Iida's research and investigation to date and materials in the public domain. Moreover, no discovery or claim construction has taken place yet in this case. As a result, Professor Iida reserves the right to amend these preliminary infringement contentions to the full extent consistent with the Court's Rules and Orders with additional or different theories and/or additional or different evidence based on his review of Intel's technical documents, Intel's document production and discovery responses, positions taken by Intel regarding non-infringement, invalidity, or claim construction, the Court's claim construction and other rulings, and his continuing investigation.

A. Asserted Claims of the ’737 Patent

The following table lists the ’737 patent and identifies the Asserted Claims.

Patent Number	Asserted Claims
6,812,737	1, 2, 7, 8, 13, 14, and 15

B. Infringing Products

Based on public information available to Professor Iida and without the benefit of discovery, the following table lists all products currently accused of infringing the ’737 patent and identifies the Asserted Claims infringed by each product.

Accused Products	**Asserted Claims**
Stratix II FPGA products including EP2S15, EP2S30, EP2S60, EP2S90, EP2S130, and EP2S180	1, 2, 7, 8, 13, 14, and 15
Stratix III E FPGA products including EP3SE50, EP3SE80, EP3SE110, and EP3SE260	1, 2, 7, 8, 13, 14, and 15
Stratix III L FPGA products including EP3SL50, EP3SL70, EP3SL110, EP3SL150, EP3SL200, and EP3SL340	1, 2, 7, 8, 13, 14, and 15
Arria GX FPGA products including EP1AGX20, EP1AGX35, EP1AGX50, EP1AGX60 and EP1AGX90	1, 2, 7, 8, 13, 14, and 15
Stratix IV GT FPGA products including EP4S40G2, EP4S40G5, EP4S100G2, EP4S100G3, EP4S100G4, and EP4S100G5	1, 2, 7, 8, 13, 14, and 15
Stratix IV GX FPGA products including EP4SGX70, EP4SGX110, EP4SGX180, EP4SGX230, EP4SGX290, EP4SGX360, and EP4SGX530	1, 2, 7, 8, 13, 14, and 15
Stratix IV E FPGA products including EP4SE230, EP4SE360, EP4SE530, and EP4SE820	1, 2, 7, 8, 13, 14, and 15
Arria II GX FPGA products including EP2AGX45, EP2AGX65, EP2AGX95, EP2AGX125, EP2AGX190, and EP2AGX260	1, 2, 7, 8, 13, 14, and 15
Arria II GZ FPGA products including EP2AGZ225, EP2AGZ300, and EP2AGZ350	1, 2, 7, 8, 13, 14, and 15
Stratix V E FPGA products including 5SEE9 and 5SEEB	1, 2, 7, 8, 13, 14, and 15
Stratix V GS FPGA products including 5SGSD3, 5SGSD4, 5SGSD5, 5SGSD6, and 5SGSD8	1, 2, 7, 8, 13, 14, and 15
Stratix V GX FPGA products including 5SGXA3, 5SGXA4, 5SGXA5, 5SGXA7, 5SGXA9, 5SGXAB, 5SGXB5, 5SGXB6, 5SGXB9, and 5SGXBB	1, 2, 7, 8, 13, 14, and 15

Arria V GT FPGA products including 5AGTC3, 5AGTC7, 5AGTD3, and 5AGTD7	1, 2, 7, 8, 13, 14, and 15
Arria V GX FPGA products including 5AGXA1, 5AGXA3, 5AGXA5, 5AGXA7, 5AGXB1, 5AGXB3, 5AGXB5, and 5AGXB7	1, 2, 7, 8, 13, 14, and 15
Arria V GZ FPGA products including 5AGZE1, 5AGZE3, 5AGZE5, and 5AGZE7	1, 2, 7, 8, 13, 14, and 15
Arria V ST SoC products including 5ASTD3 and 5ASTD5	1, 2, 7, 8, 13, 14, and 15
Arria V SX SoC products including 5ASXB3 and 5ASXB5	1, 2, 7, 8, 13, 14, and 15
Cyclone V E FPGA products including 5CEA2, 5CEA4, 5CEA5, 5CEA7, and 5CEA9	1, 2, 7, 8, 13, 14, and 15
Cyclone V GT FPGA products including 5CGTD5, 5CGTD7, and 5CGTD9	1, 2, 7, 8, 13, 14, and 15
Cyclone V GX FPGA products including 5CGXC3, 5CGXC4, 5CGXC5, 5CGXC7, and 5CGXC9	1, 2, 7, 8, 13, 14, and 15
Cyclone V SE SoC products including 5CSEA2, 5CSEA4, 5CSEA5, and 5CSEA6	1, 2, 7, 8, 13, 14, and 15
Cyclone V ST SoC products including 5CSTD5 and 5CSTD6	1, 2, 7, 8, 13, 14, and 15
Cyclone V SX SoC products including 5CSXC2, 5CSXC4, 5CSXC5, and 5CSXC6	1, 2, 7, 8, 13, 14, and 15
Stratix 10 DX SoC FPGA products including DX 1100, DX 2100, and DX 2800	1, 2, 7, 8, 13, 14, and 15
Stratix 10 GX FPGA products including GX 400, GX 650, GX 850, GX 1100, GX 1650, GX 2100, GX 2500, GX 2800, GX 1660, GX 2110, and GX 10M	1, 2, 7, 8, 13, 14, and 15
Stratix 10 MX FPGA products including MX 1650 and MX 2100	1, 2, 7, 8, 13, 14, and 15
Stratix 10 SX SoC products including SX 400, SX 650, SX 850, SX 1100, SX 1650, SX 2100, SX 2500, and SX 2800	1, 2, 7, 8, 13, 14, and 15
Stratix 10 TX FPGA products including TX 400, TX 850, TX 1100, TX 1650, TX 2100, TX 2500, and TX 2800	1, 2, 7, 8, 13, 14, and 15
Stratix NX FPGA products including NX 2100	1, 2, 7, 8, 13, 14, and 15
Arria 10 GT FPGA products including GT 900 and GT 1150	1, 2, 7, 8, 13, 14, and 15
Arria 10 GX FPGA products including GX 160, GX 220, GX 270, GX 320, GX 480, GX 570, GX 660, GX 900, and GX 1150	1, 2, 7, 8, 13, 14, and 15
Arria 10 SX SoC products including SX 160, SX 220, SX 270, SX 320, SX 480, SX 570, and SX 660	1, 2, 7, 8, 13, 14, and 15
Cyclone 10 GX FPGA products including 10CX085, 10CX105, 10CX150, and 10CX220	1, 2, 7, 8, 13, 14, and 15
Agilex F-Series FPGA and SoC FPGA products including AGF 004, AGF 006, AGF 008, AGF 012, AGF 014, AGF 022, and AGF 027	1, 2, 7, 8, 13, 14, and 15

Agilex I-Series SoC FPGA products including AGI 022 and AGI 027	1, 2, 7, 8, 13, 14, and 15
Agilex M-Series FPGA products including AGM 032 and AGM 039	1, 2, 7, 8, 13, 14, and 15

Professor Iida reserves the right to identify additional infringing Intel products, systems, or services to the full extent consistent with the Court's Rules and Orders based on his review of Intel's technical documents, Intel's document production and discovery responses, positions taken by Intel regarding non-infringement, invalidity, or claim construction, the Court's claim construction and other rulings, and his continuing investigation.

C. Infringement Theories.

As more fully detailed in the Complaint (Dkt. #1), Professor Iida asserts that Intel has directly, indirectly, and willfully infringed the Asserted Claims of the '737 patent. Professor Iida's Complaint and any amendments or supplements thereto, including the allegations and underlying facts demonstrating Intel's direct infringement, induced infringement, and willful infringement, are incorporated by reference as if fully set forth herein.

Professor Iida reserves the right to amend or supplement his infringement theories as the case progresses to the full extent consistent with the Court's Rules and Orders based on his review of Intel's technical documents, Intel's document production and discovery responses, positions taken by Intel regarding non-infringement, invalidity, or claim construction, the Court's claim construction and other rulings, and his continuing investigation.

D. Claim Chart.

All of the Accused Products employ the same infringing components: Adaptive Logic Modules. As a result, one representative claim chart identifying where each element of each

Asserted Claim is found within an exemplary Accused Product (Stratix IV) is attached hereto as Exhibit A and is incorporated by reference as if fully set forth herein.

Each element of each Asserted Claim is considered to be literally present and is also present, alternatively, under the doctrine of equivalents in the event such element is not found to be literally present.

Nothing in this claim chart is intended to prevent Professor Iida from presenting additional evidence of infringement at trial.

Professor Iida reserves the right to amend or supplement this claim chart to the full extent consistent with the Court's Rules and Orders based on his review of Intel's technical documents, Intel's document production and discovery responses, positions taken by Intel regarding non-infringement, invalidity, or claim construction, the Court's claim construction and other rulings, and his continuing investigation.

E. The Priority Date to Which Each Asserted Claim Is Entitled.

Patent Number	Asserted Claims	Priority Date
6,812,737	1, 2, 7, 8, 13, 14, and 15	June 29, 2001

F. Document Production Regarding Conception and Reduction to Practice

Professor Iida produces herewith documents evidencing his conception and reduction to practice of the claimed invention. These documents can be found within the Bates-label range P-00001 – P-00135. Please note that the English language bookmarks embedded within the PDF document served herewith have been added for convenience.

G. File Histories

Professor Iida produces herewith a copy of the file history for the ’737 patent. This file history can be found within the Bates-label range P-00136 – P-00376.

Dated: September 20, 2022

Respectfully submitted,

/s/ *Joshua R. Slavitt*
Joshua R. Slavitt*
Pennsylvania State Bar No. 63139
SLAVITT IP LAW, LLC
535 Hamilton Road
Merion Station, PA 19066
Phone: (215) 880-2569
Email: josh@slavittiplaw.com

Travis C. Barton
Texas State Bar No. 00790276
Richard D. Milvenan
Texas State Bar No. 14171800
McGINNIS & LOCHRIDGE LLP
1111 W. 6th Street, Bldg. B, Suite 400
Austin, TX 78703
Phone: (512) 495-6005
Facsimile: (512) 505-6305
Email: tcbarton@mcginnislaw.com
Email: rmilvenan@mcginnislaw.com

Jacob C. Cohn*
Pennsylvania State Bar No. 54139
Ilan Rosenberg*
Pennsylvania State Bar No. 89668
GORDON REES SCULLY MANSUKHANI LLP
1717 Arch Street, Suite 610
Philadelphia, PA 19103
Phone: (215) 561-2300
Email: jcohn@grsm.com
Email: irosenberg@grsm.com

**Admitted pro hac vice*

Attorneys for Plaintiff,
Professor Masahiro Iida

CERTIFICATE OF SERVICE

The undersigned hereby certifies that on September 20, 2022, a true and correct copy of the foregoing document as well as the Documents P-00001 – P-00376 were served via email upon all counsel of record.

By: /s/Joshua R. Slavitt
Joshua R. Slavitt

EXHIBIT A

Professor Masahiro Iida v. Intel Corporation – Civil Action No. 6:22-cv-00662 (E.D. Tex.)

US Patent No. 6,812,737 (`737 Patent) - Claims 1, 2, 7, 8, 13, 14, 15

Priority Date: June 29, 2001

PROGRAMMABLE LOGIC CIRCUIT DEVICE HAVING LOOK UP TABLE ENABLING TO REDUCE IMPLEMENTATION AREA

(12) **United States Patent**
Sueyoshi et al.

(10) **Patent No.:** **US 6,812,737 B2**
(45) **Date of Patent:** **Nov. 2, 2004**

(54) **PROGRAMMABLE LOGIC CIRCUIT DEVICE HAVING LOOK UP TABLE ENABLING TO REDUCE IMPLEMENTATION AREA**

(75) Inventors: **Toshinori Sueyoshi**, Kumamoto (JP); **Masahiro Iida**, Fujisawa (JP)

(73) Assignee: **Semiconductor Technology Academic Research Center**, Kanagawa (JP)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **10/183,590**

(22) Filed: **Jun. 28, 2002**

(65) **Prior Publication Data**

US 2003/0001615 A1 Jan. 2, 2003

(30) **Foreign Application Priority Data**

Jun. 29, 2001 (JP) 2001-199644

(51) **Int. Cl.**[7] **H03K 19/173**
(52) **U.S. Cl.** **326/38**; 326/39; 326/40
(58) **Field of Search** 329/37–47

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,706,216 A 11/1987 Carter
4,870,302 A 9/1989 Freeman
5,442,306 A 8/1995 Woo
5,778,439 A 7/1998 Trimberger et al.
5,905,385 A 5/1999 Sharpe-Geisler
5,909,126 A * 6/1999 Cliff et al. 326/41
5,999,015 A * 12/1999 Cliff et al. 326/39
6,323,677 B1 * 11/2001 Lane et al. 326/37
6,476,636 B1 * 11/2002 Lien et al. 326/41

FOREIGN PATENT DOCUMENTS

JP 03-063846 3/1991
JP 08-237109 9/1996

OTHER PUBLICATIONS

Rose, Jonathoan, et al., "Architecture of Field–Programmable Gate Arrays: The Effect of Logic Block Functionality on Area Efficiency", IEEE J. Solid–State Circuits, vol. 25, No. 5, pp. 1217–1225, Oct. 1990.
Rose, Jonathan, et al., "The Effect of Logic Block Architecture on FPGA Performance", IEEE J. Solid–State Circuits, vol. 27, No. 3., pp. 281–287, Mar. 1992.
Ahmed, Elias et al., "The Effect of LUT and Cluster Size on Deep–Submicron FPGA Performance and Density", FPGA 2000, Monterey, CA USA, 2000.

* cited by examiner

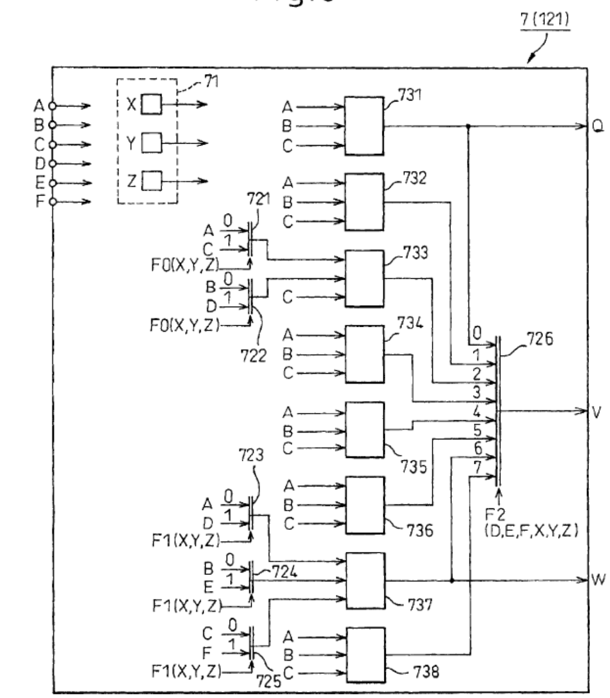
Primary Examiner—Anh Q Tran
(74) *Attorney, Agent, or Firm*—Staas & Halsey LLP

(57) **ABSTRACT**

A programmable logic circuit device has a plurality of logic blocks, a plurality of routing wires, a plurality of switch circuits, a plurality of connection blocks, and an I/O block performing an input/output operation with external equipment. The routing wires are connected to each of the logic blocks, the switch circuits are provided at an intersection of each of the routing wires, and the connection blocks are provided between an I/O line of each of the logic blocks and each of the routing wires. Each of the logic blocks has a look up table of M inputs and N outputs, which has a plurality of LUT units; and an internal configuration control circuit controlling an internal configuration of the plurality of LUT units.

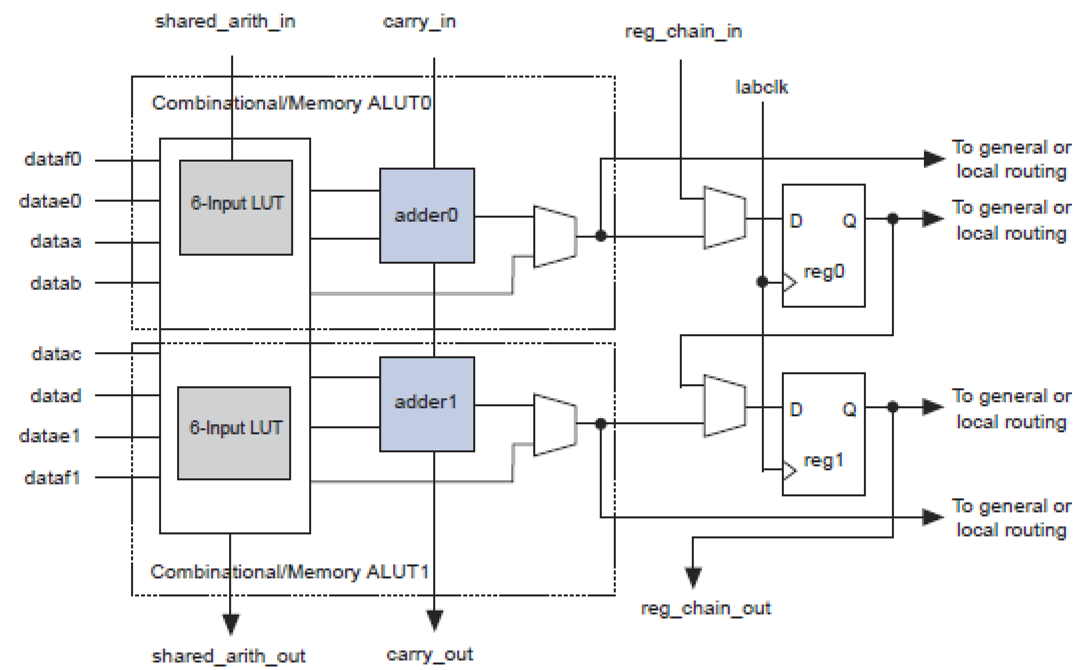
16 Claims, 14 Drawing Sheets

Fig.6



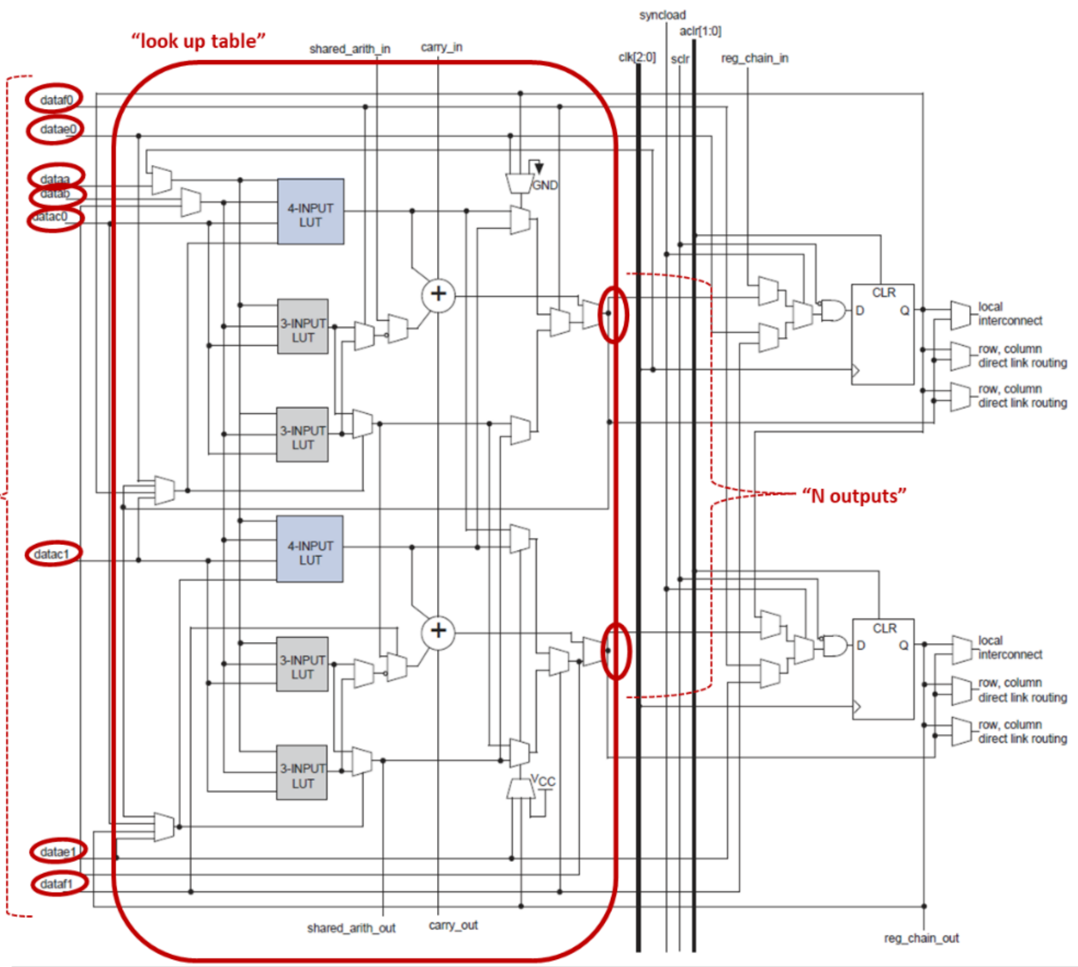
List of Cited Documents

1) (“***Stratix IV HB v1***”). Stratix IV Device Handbook, Volume 1, SIV5V1-4.0, November 2009.
2) (“***FPGA Architecture WP2006***”). “FPGA Architecture,” Altera White Paper, WP-01003-1.0 (2006)
3) *(“**Hutton2004**”).* M. Hutton, et al. “Fracturable FPGA Logic Elements” (2004)
4) (“***Lewis2003***”). D. Lewis, et al. “The Stratix Routing and Logic Architecture” 12-20. DOI:10.1145/611817.611821 (2003)
5) (“***Lewis2005***”). D. Lewis, et al. “The Stratix II Logic and Routing Architecture” FPGA 2005: ACM Symposium on FPGAs, 14-20. https://doi.org/10.1145/1046192.1046195 (2005)

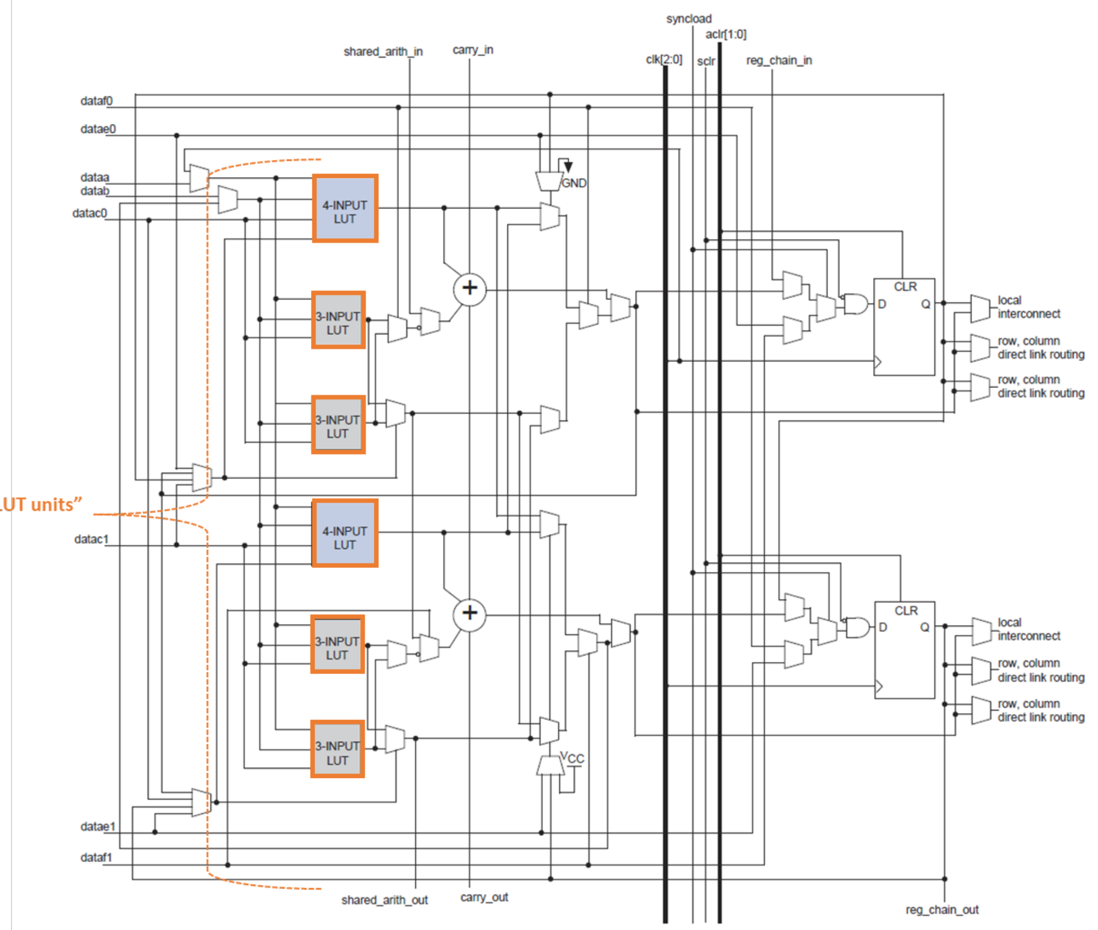
#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
1pre	1. A **look up table of M inputs and N outputs**, comprising:	**Figure 2–5.** High-Level Block Diagram of the Stratix IV ALM  ***Stratix IV HB v1 – p.2-6***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		Adaptive Logic Modules The ALM is the basic building block of logic in the Stratix IV architecture. It provides advanced features with efficient logic usage. Each ALM contains a variety of LUT-based resources that can be divided between two combinational adaptive LUTs (ALUTs) and two registers. With up to eight inputs for the two combinational ALUTs, one ALM can implement various combinations of two functions. This adaptability allows an ALM to be completely backward-compatible with four-input LUT architectures. One ALM can also implement any function with up to six inputs and certain seven-input functions. In addition to the adaptive LUT-based resources, each ALM contains two programmable registers, two dedicated full adders, a carry chain, a shared arithmetic chain, and a register chain. Through these dedicated resources, an ALM can efficiently implement various arithmetic functions and shift registers. Each ALM drives all types of interconnects: local, row, column, carry chain, shared arithmetic chain, register chain, and direct link. Figure 2–5 shows a high-level block diagram of the Stratix IV ALM. ***Stratix IV HB v1 – p.2-5***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		One ALM contains two programmable registers. Each register has data, clock, clock enable, synchronous and asynchronous clear, and synchronous load and clear inputs. Global signals, general-purpose I/O pins, or any internal logic can drive the register's clock and clear-control signals. Either general-purpose I/O pins or internal logic can drive the clock enable. For combinational functions, the register is bypassed and the output of the LUT drives directly to the outputs of an ALM. Each ALM has two sets of outputs that drive the local, row, and column routing resources. The LUT, adder, or register outputs can drive these output drivers (refer to Figure 2–6). For each set of output drivers, two ALM outputs can drive column, row, or direct-link routing connections. One of these ALM outputs can also drive local interconnect resources. This allows the LUT or adder to drive one output while the register drives another output. ***Stratix IV HB v1 – p.2-7***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		Figure 2–6. Stratix IV ALM Connection Details  ***Stratix IV HB v1 – p.2-7***

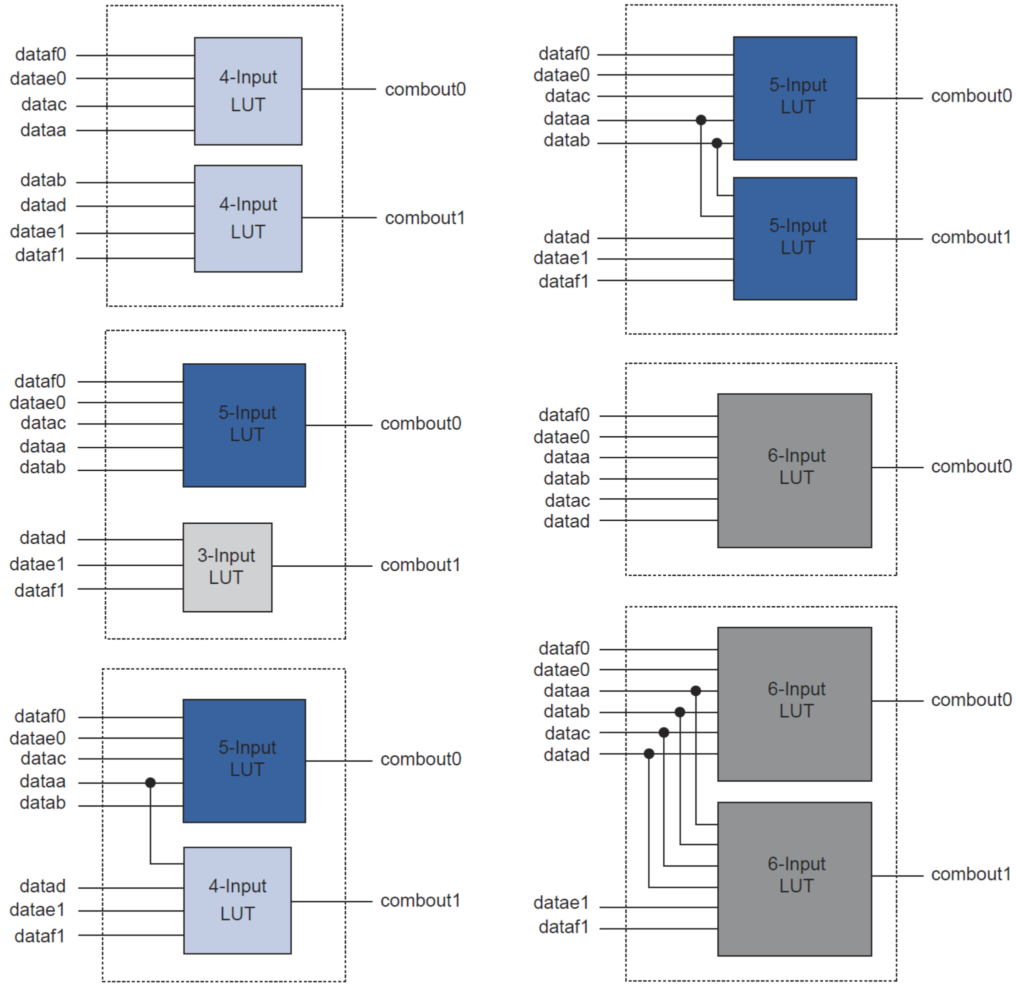
#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		Normal Mode Normal mode is suitable for general logic applications and combinational functions. In this mode, up to eight data inputs from the LAB local interconnect are inputs to the combinational logic. Normal mode allows two functions to be implemented in one Stratix IV ALM, or a single function of up to six inputs. The ALM can support certain combinations of completely independent functions and various combinations of functions that have common inputs. ***Stratix IV HB v1 – p.2-8***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
1A	a plurality of **LUT units**; and	**Figure 2–6.** Stratix IV ALM Connection Details  ***Stratix IV HB v1 – p.2-7***

1B	an **internal configuration control circuit** controlling an internal **configuration of said plurality of LUT units**, wherein said internal configuration control circuit comprises	**ALM Operating Modes** The Stratix IV ALM operates in one of the following modes: ■ Normal ■ Extended LUT ■ Arithmetic ■ Shared Arithmetic ■ LUT-Register Each mode uses ALM resources differently. In each mode, eleven available inputs to an ALM—the eight data inputs from the LAB local interconnect, carry-in from the previous ALM or LAB, the shared arithmetic chain connection from the previous ALM or LAB, and the register chain connection—are directed to different destinations to implement the desired logic function. LAB-wide signals provide clock, asynchronous clear, synchronous clear, synchronous load, and clock enable control for the register. These LAB-wide signals are available in all ALM modes. For more information about the LAB-wide control signals, refer to "LAB Control Signals" on page 2–4. The Quartus II software and supported third-party synthesis tools, in conjunction with parameterized functions such as the library of parameterized modules (LPM) functions, automatically choose the appropriate mode for common functions such as counters, adders, subtractors, and arithmetic functions. ***Stratix IV HB v1 – p.2-8***

		Normal Mode Normal mode is suitable for general logic applications and combinational functions. In this mode, up to eight data inputs from the LAB local interconnect are inputs to the combinational logic. Normal mode allows two functions to be implemented in one Stratix IV ALM, or a single function of up to six inputs. The ALM can support certain combinations of completely independent functions and various combinations of functions that have common inputs. ***Stratix IV HB v1 – p.2-8***

Figure 2–7. ALM in Normal Mode *(Note 1)*



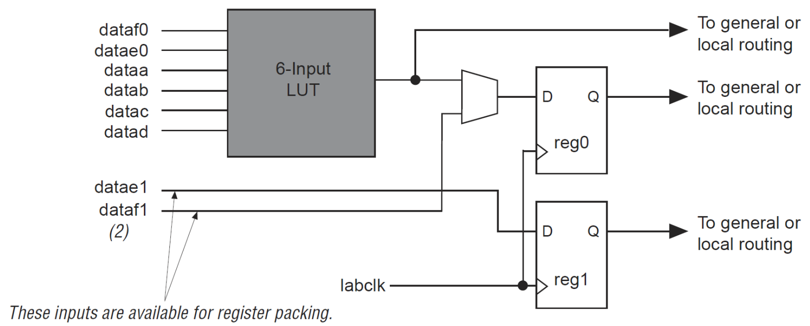
Note to Figure 2–7:

(1) Combinations of functions with fewer inputs than those shown are also supported. For example, combinations of functions with the following number of inputs are supported: 4 and 3, 3 and 3, 3 and 2, and 5 and 2.

Stratix IV HB v1 – p.2-9

		Normal mode provides complete backward-compatibility with four-input LUT architectures. For the packing of 2 five-input functions into one ALM, the functions must have at least two common inputs. The common inputs are `dataa` and `datab`. The combination of a four-input function with a five-input function requires one common input (either `dataa` or `datab`). ***Stratix IV HB v1 – p.2-9*** In the case of implementing 2 six-input functions in one ALM, four inputs must be shared and the combinational function must be the same. In a sparsely used device, functions that could be placed in one ALM may be implemented in separate ALMs by the Quartus II software to achieve the best possible performance. As a device begins to fill up, the Quartus II software automatically uses the full potential of the Stratix IV ALM. The Quartus II Compiler automatically searches for functions using common inputs or completely independent functions to be placed in one ALM to make efficient use of device resources. In addition, you can manually control resource usage by setting location assignments. You can implement any six-input function using inputs `dataa`, `datab`, `datac`, `datad`, and either `datae0` and `dataf0` or `datae1` and `dataf1`. If you use `datae0` and `dataf0`, the output is driven to `register0`, and/or `register0` is bypassed and the data drives out to the interconnect using the top set of output drivers (refer to Figure 2–8). If you use `datae1` and `dataf1`, the output either drives to `register1` or bypasses `register1` and drives to the interconnect using the bottom set of output drivers. The Quartus II Compiler automatically selects the inputs to the LUT. ALMs in normal mode support register packing. ***Stratix IV HB v1 – p.2-10***

Figure 2–8. Input Function in Normal Mode *(Note 1)*



Notes to Figure 2–8:

(1) If `datae1` and `dataf1` are used as inputs to a six-input function, `datae0` and `dataf0` are available for register packing.

(2) The `dataf1` input is available for register packing only if the six-input function is unregistered.

Stratix IV HB v1 – p.2-10

Extended LUT Mode

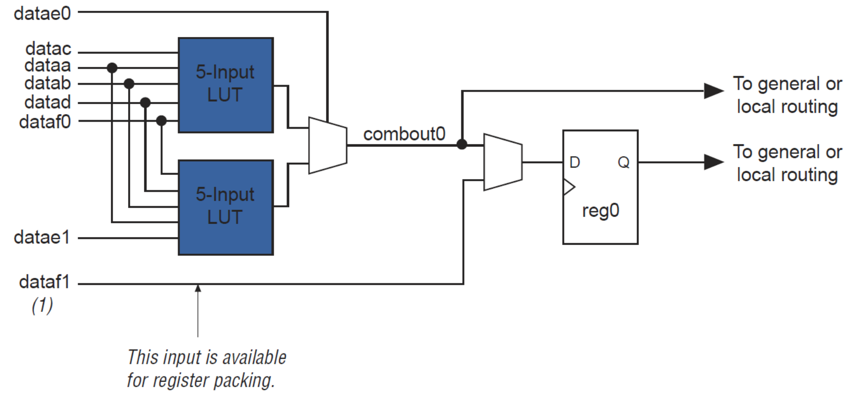
Use extended LUT mode to implement a specific set of seven-input functions. The set must be a 2-to-1 multiplexer fed by two arbitrary five-input functions sharing four inputs. Figure 2–9 shows the template of supported seven-input functions using extended LUT mode. In this mode, if the seven-input function is unregistered, the unused eighth input is available for register packing.

Functions that fit into the template shown in Figure 2–9 occur naturally in designs. These functions often appear in designs as “if-else” statements in Verilog HDL or VHDL code.

Stratix IV HB v1 – p.2-10

Professor Masahiro Iida v. Intel Corporation – Civil Action No. 6:22-cv-00662 (E.D. Tex.)

Figure 2–9. Template for Supported Seven-Input Functions in Extended LUT Mode



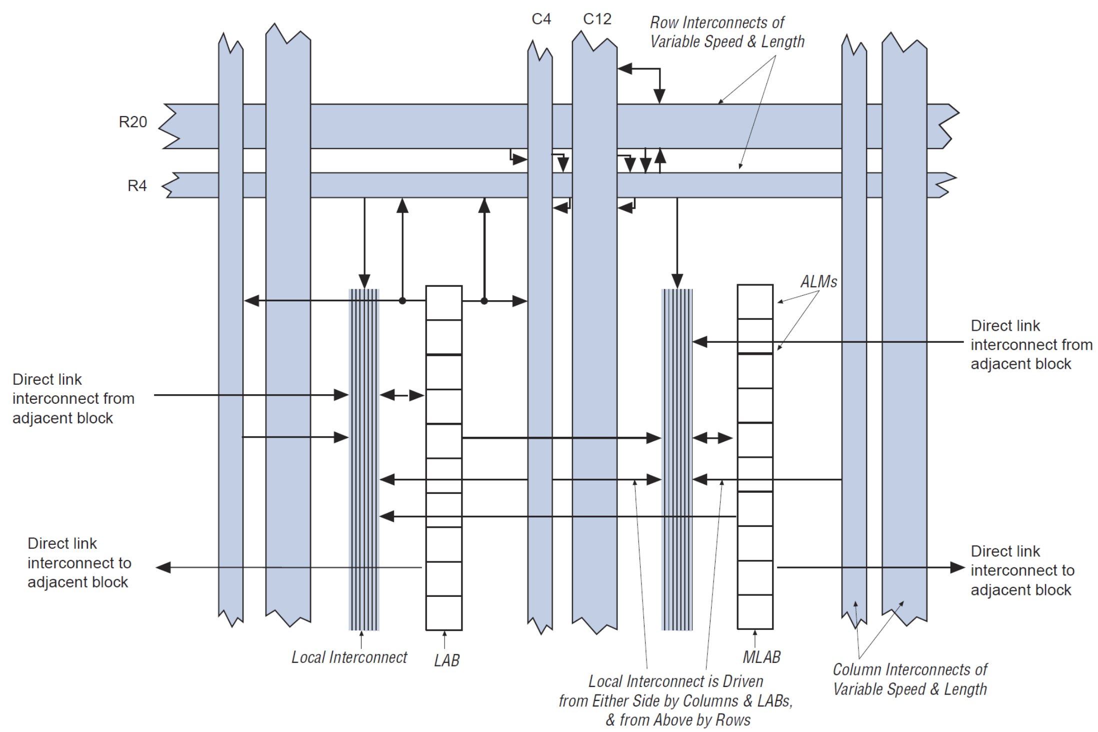
Note to Figure 2–9:

(1) If the seven-input function is unregistered, the unused eighth input is available for register packing. The second register, `reg1`, is not available.

Stratix IV HB v1 – p.2-11

		Overview This chapter describes supported configuration schemes for Stratix IV devices, instructions about how to execute the required configuration schemes, and the necessary pin settings. Stratix IV devices use SRAM cells to store configuration data. As SRAM is volatile, you must download configuration data to the Stratix IV device each time the device powers up. You can configure Stratix IV devices using one of four configuration schemes: ■ Fast passive parallel (FPP) ■ Fast active serial (AS) ■ Passive serial (PS) ■ Joint Test Action Group (JTAG) All configuration schemes use either an external controller (for example, a MAX® II device or microprocessor), a configuration device, or a download cable. For more information, refer to "Configuration Features" on page 10–4. ***Stratix IV HB v1 – p.10-1***

Figure 2–1. Stratix IV LAB Structure



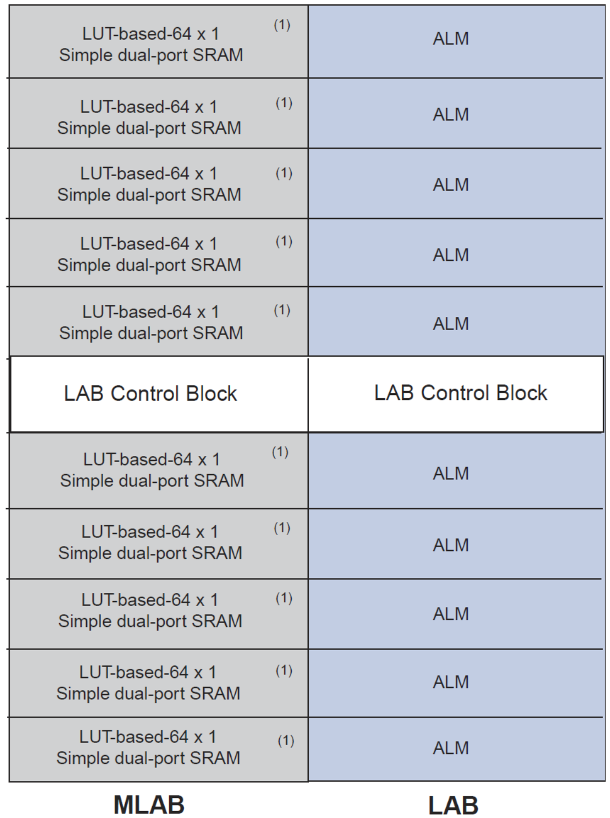
Stratix IV HB v1 – p.2-2

The LAB of the Stratix IV device has a derivative called memory LAB (MLAB), which adds look-up table (LUT)-based SRAM capability to the LAB, as shown in Figure 2–2. The MLAB supports a maximum of 640 bits of simple dual-port static random access memory (SRAM). You can configure each ALM in an MLAB as either a 64 × 1 or a 32 × 2 block, resulting in a configuration of either a 64 × 10 or a 32 × 20 simple dual-port SRAM block. MLAB and LAB blocks always coexist as pairs in all Stratix IV families. MLAB is a superset of the LAB and includes all LAB features.

The MLAB is described in detail in the *TriMatrix Embedded Memory Blocks in Stratix IV Devices* chapter.

Stratix IV HB v1 – p.2-2

Figure 2–2. Stratix IV LAB and MLAB Structure



Note to Figure 2–2:

(1) You can use the MLAB ALM as a regular LAB ALM or configure it as a dual-port SRAM, as shown.

Stratix IV HB v1 – p. 2-3

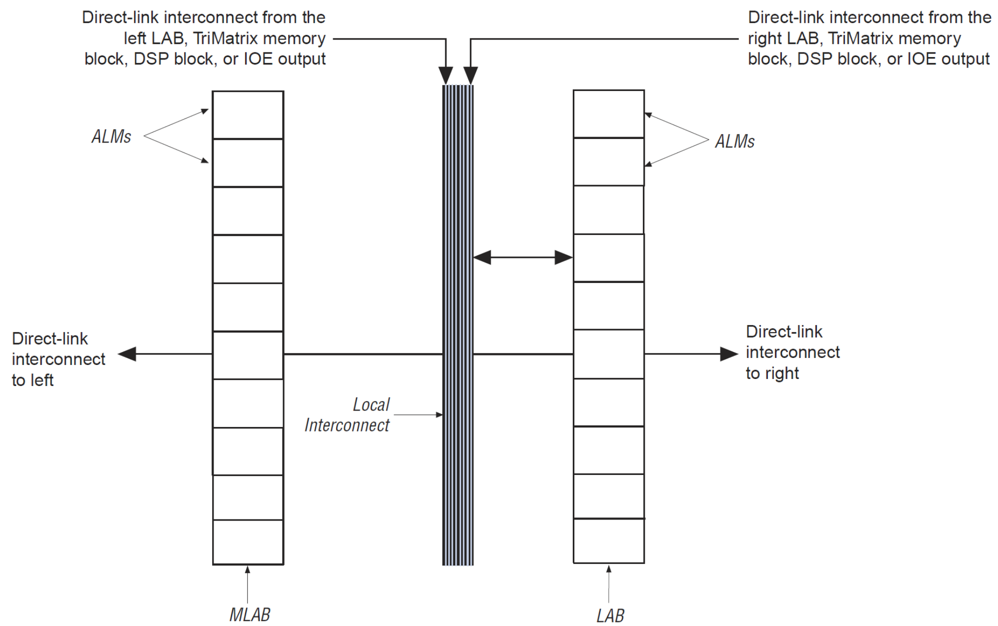
LAB Interconnects

The LAB local interconnect can drive ALMs in the same LAB. It is driven by column and row interconnects and ALM outputs in the same LAB. Neighboring LABs/MLABs, M9K RAM blocks, M144K blocks, or DSP blocks from the left or right can also drive the LAB's local interconnect through the direct link connection. The direct link connection feature minimizes the use of row and column interconnects, providing higher performance and flexibility. Each LAB can drive 30 ALMs through fast-local and direct-link interconnects.

Stratix IV HB v1 – p. 2-3

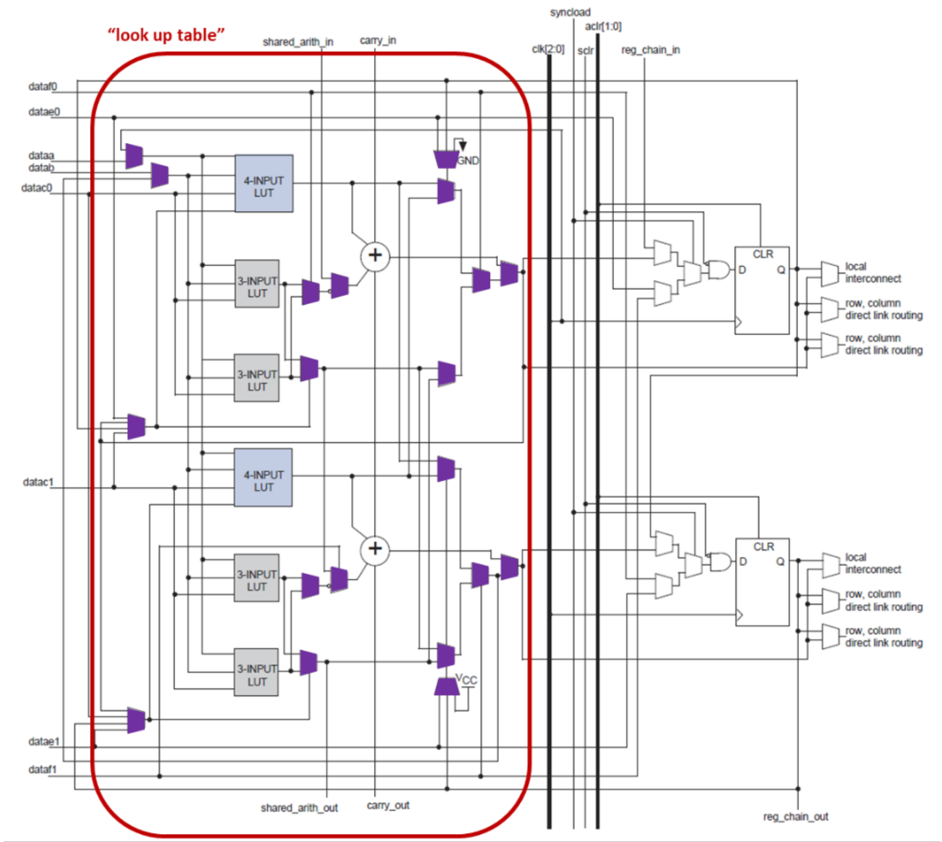
Figure 2–3 shows the direct-link connection.

Figure 2–3. Direct-Link Connection



Stratix IV HB v1 – p. 2-4

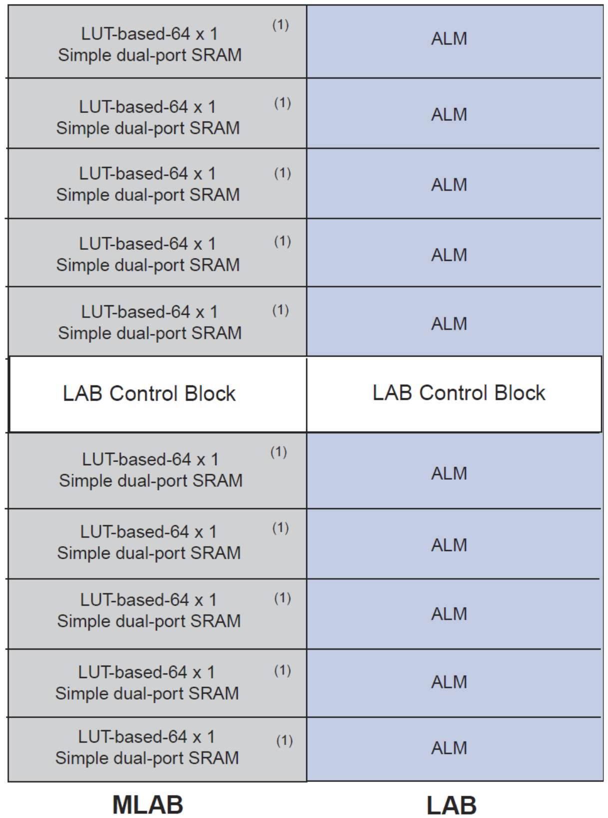
<table>
<tr>
<td>1C</td>
<td>a plurality of selectors selecting I/O signals of said plurality of LUT units, and</td>
<td>
Figure 2–6. Stratix IV ALM Connection Details



Stratix IV HB v1 – p.2-7
</td>
</tr>
</table>

1D	a **selector control circuit** having a **memory**, controlling said plurality of selectors in accordance with **data stored in said memory**, and **defining the internal configuration** of said plurality of LUT units.	***See claim elements 1A and 1B, at least showing the various internal LUT configurations.*** **Overview** This chapter describes supported configuration schemes for Stratix IV devices, instructions about how to execute the required configuration schemes, and the necessary pin settings. Stratix IV devices use SRAM cells to store configuration data. As SRAM is volatile, you must download configuration data to the Stratix IV device each time the device powers up. You can configure Stratix IV devices using one of four configuration schemes: ■ Fast passive parallel (FPP) ■ Fast active serial (AS) ■ Passive serial (PS) ■ Joint Test Action Group (JTAG) All configuration schemes use either an external controller (for example, a MAX® II device or microprocessor), a configuration device, or a download cable. For more information, refer to "Configuration Features" on page 10–4. ***Stratix IV HB v1 – p.10-1*** The LAB of the Stratix IV device has a derivative called memory LAB (MLAB), which adds look-up table (LUT)-based SRAM capability to the LAB, as shown in Figure 2–2. The MLAB supports a maximum of 640 bits of simple dual-port static random access memory (SRAM). You can configure each ALM in an MLAB as either a 64 × 1 or a 32 × 2 block, resulting in a configuration of either a 64 × 10 or a 32 × 20 simple dual-port SRAM block. MLAB and LAB blocks always coexist as pairs in all Stratix IV families. MLAB is a superset of the LAB and includes all LAB features. The MLAB is described in detail in the *TriMatrix Embedded Memory Blocks in Stratix IV Devices* chapter. ***Stratix IV HB v1 – p. 2-2***

Figure 2–2. Stratix IV LAB and MLAB Structure



Note to Figure 2–2:

(1) You can use the MLAB ALM as a regular LAB ALM or configure it as a dual-port SRAM, as shown.

Stratix IV HB v1 – p. 2-3

Adaptive Logic Modules

The ALM is the basic building block of logic in the Stratix IV architecture. It provides advanced features with efficient logic usage. Each ALM contains a variety of LUT-based resources that can be divided between two combinational adaptive LUTs (ALUTs) and two registers. With up to eight inputs for the two combinational ALUTs, one ALM can implement various combinations of two functions. This adaptability allows an ALM to be completely backward-compatible with four-input LUT architectures. One ALM can also implement any function with up to six inputs and certain seven-input functions.

In addition to the adaptive LUT-based resources, each ALM contains two programmable registers, two dedicated full adders, a carry chain, a shared arithmetic chain, and a register chain. Through these dedicated resources, an ALM can efficiently implement various arithmetic functions and shift registers. Each ALM drives all types of interconnects: local, row, column, carry chain, shared arithmetic chain, register chain, and direct link. Figure 2–5 shows a high-level block diagram of the Stratix IV ALM.

Stratix IV HB v1 – p.2-5

One ALM contains two programmable registers. Each register has data, clock, clock enable, synchronous and asynchronous clear, and synchronous load and clear inputs. Global signals, general-purpose I/O pins, or any internal logic can drive the register's clock and clear-control signals. Either general-purpose I/O pins or internal logic can drive the clock enable. For combinational functions, the register is bypassed and the output of the LUT drives directly to the outputs of an ALM.

Each ALM has two sets of outputs that drive the local, row, and column routing resources. The LUT, adder, or register outputs can drive these output drivers (refer to Figure 2–6). For each set of output drivers, two ALM outputs can drive column, row, or direct-link routing connections. One of these ALM outputs can also drive local interconnect resources. This allows the LUT or adder to drive one output while the register drives another output.

Stratix IV HB v1 – p.2-7

This chapter describes the TriMatrix embedded memory blocks in Stratix® IV devices. TriMatrix embedded memory blocks provide three different sizes of embedded SRAM to efficiently address the needs of Stratix IV FPGA designs. TriMatrix memory includes 640-bit memory logic array blocks (MLABs), 9-Kbit M9K blocks, and 144-Kbit M144K blocks. MLABs have been optimized to implement filter delay lines, small FIFO buffers, and shift registers. You can use the M9K blocks for general purpose memory applications and the M144K blocks for processor code storage, packet buffering, and video frame buffering.

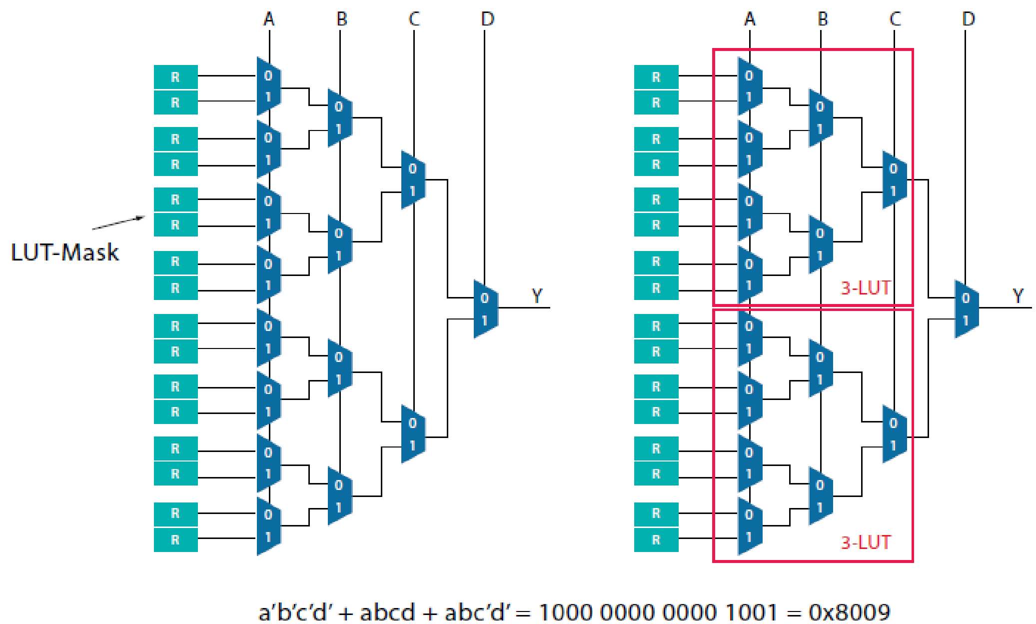
You can independently configure each embedded memory block to be a single- or dual-port RAM, FIFO buffer, ROM, or shift register using the Quartus® II MegaWizard™ Plug-In Manager. You can stitch together multiple blocks of the same type to produce larger memories with minimal timing penalty. TriMatrix memory provides up to 31,491 Kbits of embedded SRAM at up to 600 MHz operation.

Stratix IV HB v1 – p.3-1

Building Look-up Tables (LUTs)

An overview of how LUTs are built helps describe the key innovations in the ALM. A LUT is typically built out of SRAM bits to hold the configuration memory (CRAM) LUT-mask and a set of multiplexers to select the bit of CRAM that is to drive the output. To implement a k-input LUT (k-LUT)—a LUT that can implement any function of k inputs—2^k SRAM bits and a 2^k:1 multiplexer are needed. Figure 2 shows a 4-LUT, which consists of 16 bits of SRAM and a 16:1 multiplexer implemented as a tree of 2:1 multiplexers. The 4-LUT can implement any function of 4 inputs (A, B, C, D) by setting the appropriate value in the LUT-mask. To simplify the 4-LUT in Figure 2, it can also be built from two 3-LUTs connected by a 2:1 multiplexer.

Figure 2. Building a LUT



Similarly, larger LUTs can be built out of smaller ones, as shown in Figure 3. For example, a 5-LUT can be built with two 4-LUTs and a multiplexer, while a 6-LUT can be built with two 5-LUTs and a multiplexer. Technically, what matters is the total number of CRAM bits in the LUT and that they are used to implement an arbitrary function of six inputs.

FPGA Architecture WP2006 – p. 3

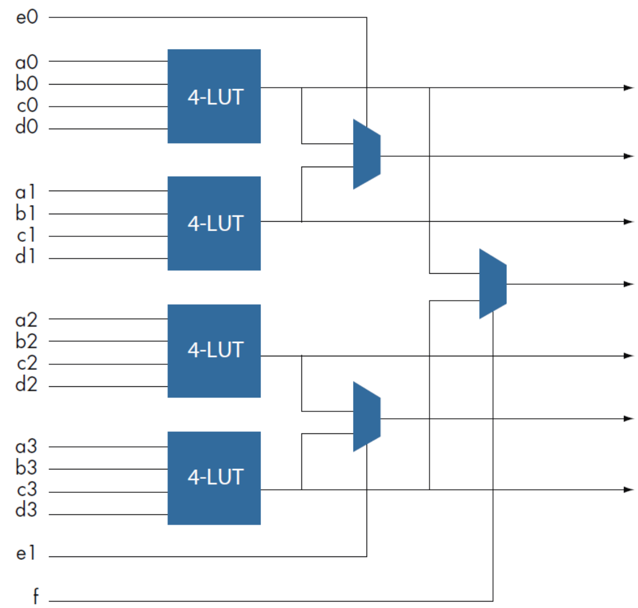
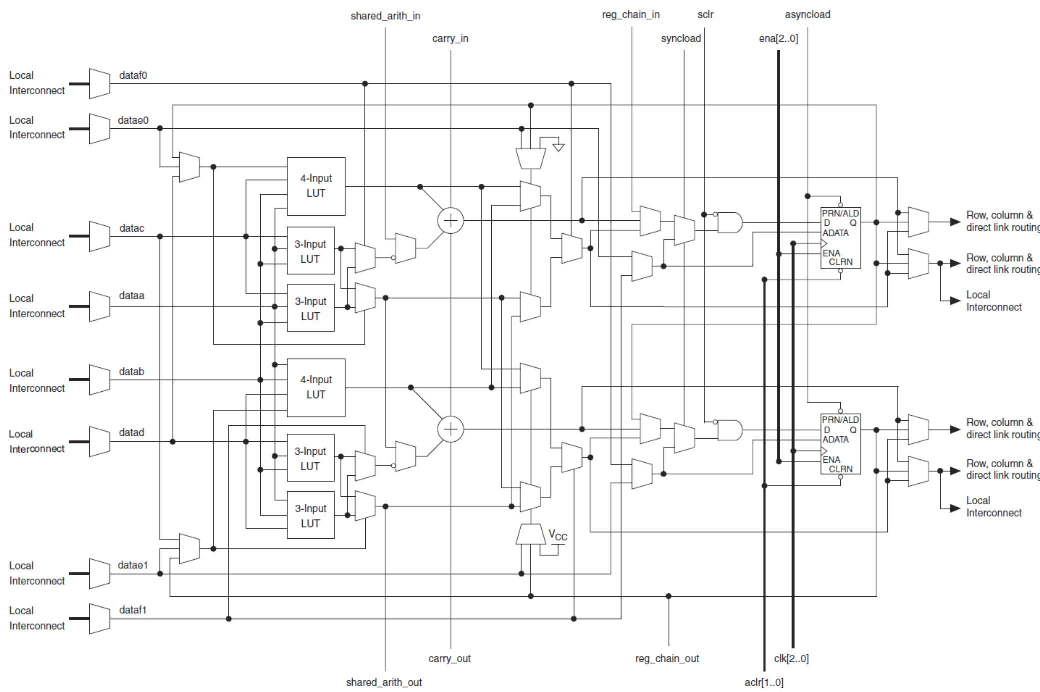
		Figure 3. Composing Larger LUTs from Smaller LUTs  Even though larger LUTs can be built from smaller LUTs, it is important to differentiate between FPGA architectures designed for 4-LUTs and for 6-LUTs. With a different size LUT as the base logic block, the number of LUTs clustered together in each architecture, the number of inputs available to the LUTs, and delay optimization through the LUTs will vary. While 6-LUTs can be built on architectures that support 4-LUTs, this structure is inefficient. For example, four 4-LUTs together with either a 4:1 multiplexer or 2 more 4-LUTs can be used to build a 6-LUT as shown in Figure 3, but the implementation uses only 6 of the 16 available inputs and creates extra delays between the various LUTs. Clearly, having the ability to build 6-input LUTs is not enough; the entire architecture needs to be optimized specifically for 6-LUTs as the base logic block. Altera was the first to offer an architecture optimized for 6-LUT performance with the Stratix II FPGA family. ***FPGA Architecture WP2006 – p. 4*** Figure 6 shows a different representation of the ALM in terms of 4-input and 3-input LUTs and multiplexers, illustrating how the LUT-mask can be fractured and shared between two different logic functions. Approximately 150,000 FPGA synthesis, placement, and routing runs were performed to determine the most cost-effective structure that would yield the performance improvement of a 6-LUT. ***FPGA Architecture WP2006 – p. 6***

Figure 6. Adaptive Logic Module (ALM) Block Diagram



FPGA Architecture WP2006 – p. 7

		The ALM Advantage The Stratix II ALM has put Altera at least one generation ahead of the competition in FPGA architecture. Introduced more than two years ago, it is significantly more flexible and, as a result, more area efficient than the recently introduced Xilinx Virtex-5 logic element (also called a LUT flipflop pair), which consists of a basic 6-LUT, carry logic, and a single register as shown in Figure 8. In comparison, the combinational logic portion of the ALM has 8 inputs and supports all 6-input functions plus many other combinations of smaller functions using its 2 outputs. The combinational logic portion of the Virtex-5 logic element, a basic 6-LUT, also has 64 bits of CRAM and two outputs like the ALM, but only contains 6 inputs and has a limited ability to implement more than one logic function. One of its outputs is the output of the 6-LUT and the other is the 5-LUT corresponding to the lower half of the configuration RAM. ***FPGA Architecture WP2006 – p. 8***

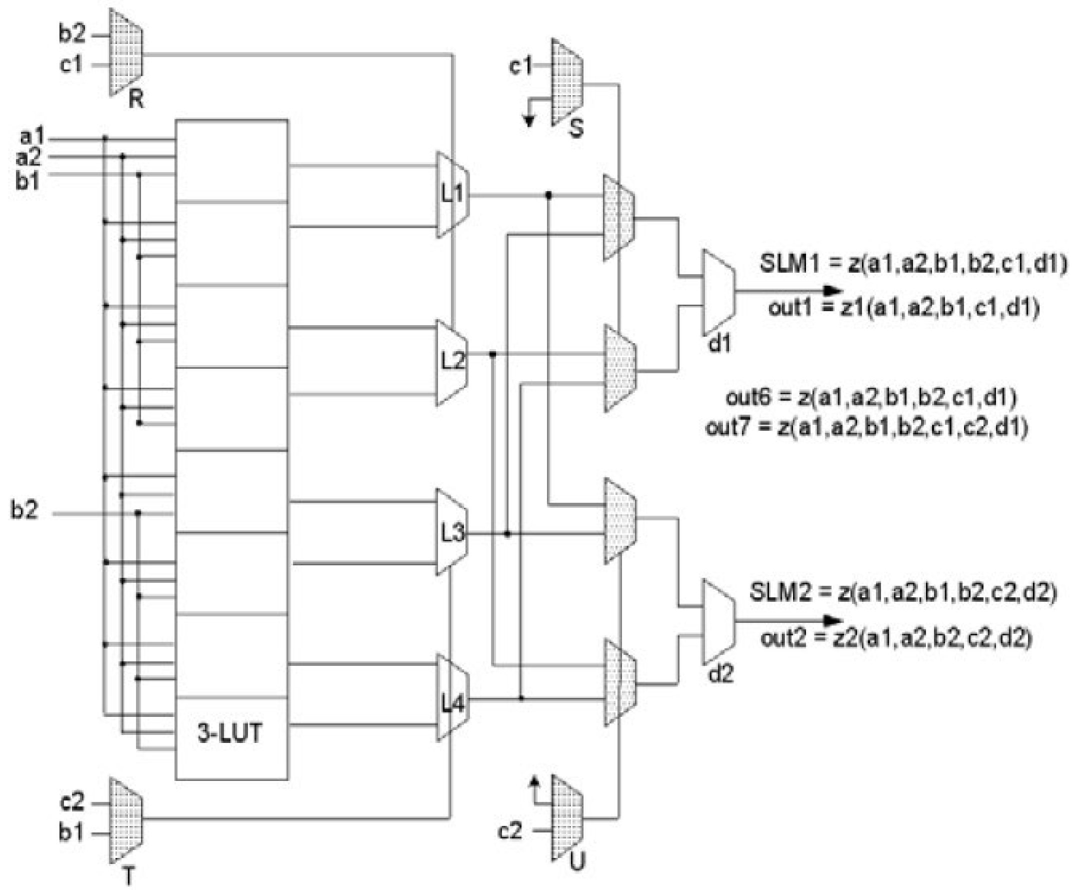
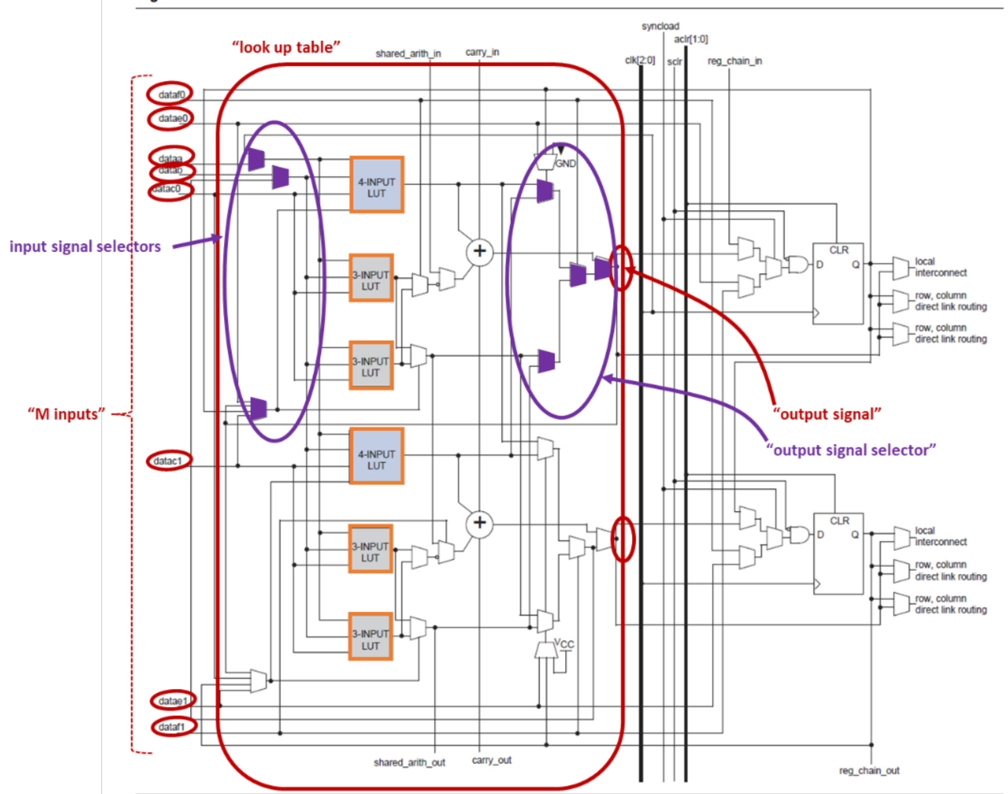


Figure 9. 6,2 ALM with 2 outputs and SLM

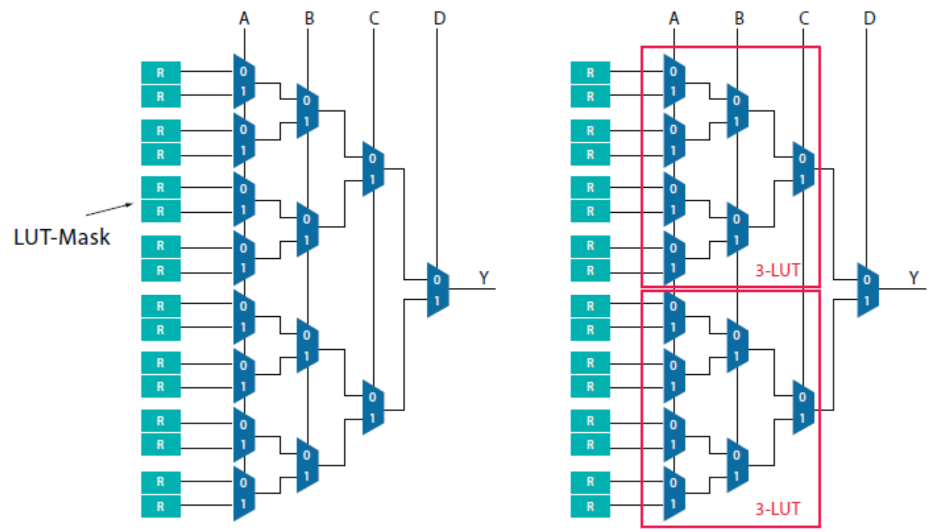
Hutton2004 – p. 5

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
2A	The look up table as claimed in claim 1, wherein said plurality of **selectors** include: an **input signal selector** provided at **an input of at least one of said LUT units** to select an input signal; and an **output signal selector** provided at an **output of said LUT units** selecting an output signal,	*Commentary: Each one of the multiplexers at the input is individually an "input signal selector." Each one of the multiplexers at the output is an "output signal selector" also a POSITA would understand that collectively the output signal selectors are a single, larger, "output signal selector".* ***See claim elements 1A, 1B and 1C.*** **Figure 2–6.** Stratix IV ALM Connection Details  ***Stratix IV HB v1 – p.2-7***

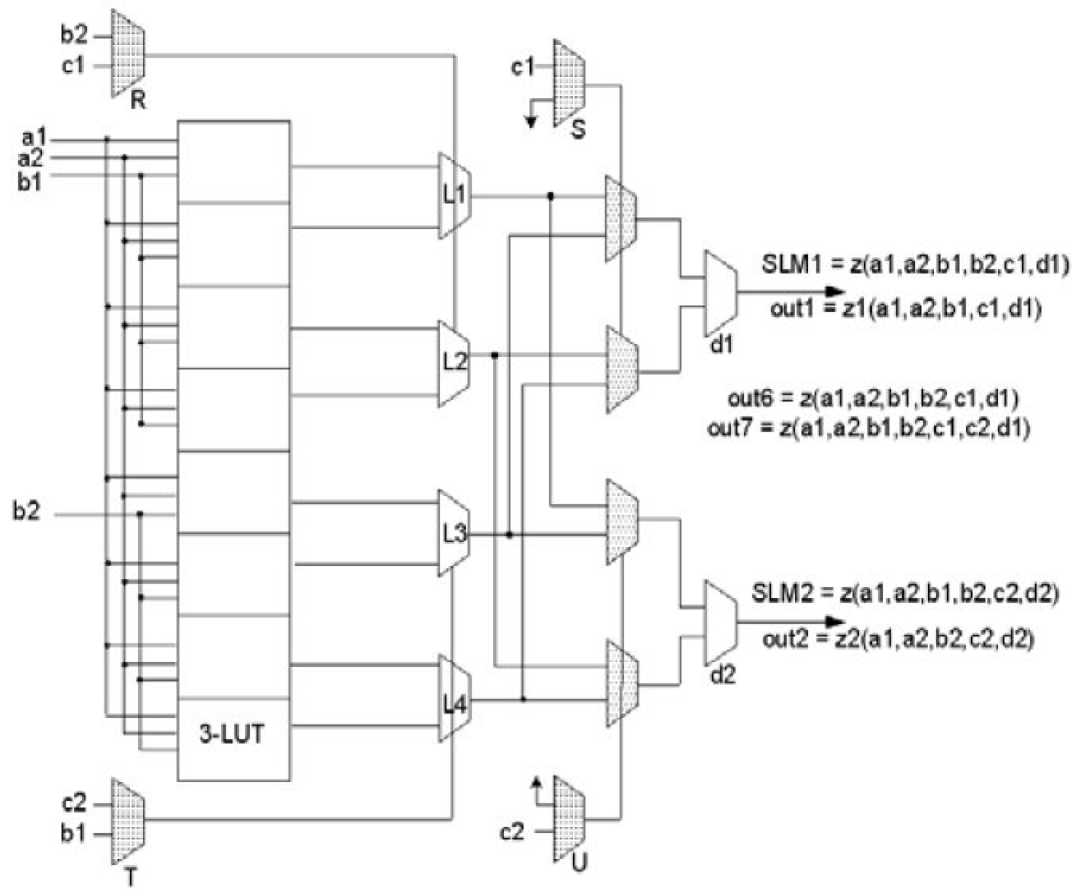
#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
2B	said input signal selector and said output signal selector being **controlled in accordance with the data stored in said memory**.	***See claim element 1D.*** **Overview** This chapter describes supported configuration schemes for Stratix IV devices, instructions about how to execute the required configuration schemes, and the necessary pin settings. Stratix IV devices use SRAM cells to store configuration data. As SRAM is volatile, you must download configuration data to the Stratix IV device each time the device powers up. You can configure Stratix IV devices using one of four configuration schemes: ■ Fast passive parallel (FPP) ■ Fast active serial (AS) ■ Passive serial (PS) ■ Joint Test Action Group (JTAG) All configuration schemes use either an external controller (for example, a MAX® II device or microprocessor), a configuration device, or a download cable. For more information, refer to "Configuration Features" on page 10–4. ***Stratix IV HB v1 – p.10-1***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		Adaptive Logic Modules The ALM is the basic building block of logic in the Stratix IV architecture. It provides advanced features with efficient logic usage. Each ALM contains a variety of LUT-based resources that can be divided between two combinational adaptive LUTs (ALUTs) and two registers. With up to eight inputs for the two combinational ALUTs, one ALM can implement various combinations of two functions. This adaptability allows an ALM to be completely backward-compatible with four-input LUT architectures. One ALM can also implement any function with up to six inputs and certain seven-input functions. In addition to the adaptive LUT-based resources, each ALM contains two programmable registers, two dedicated full adders, a carry chain, a shared arithmetic chain, and a register chain. Through these dedicated resources, an ALM can efficiently implement various arithmetic functions and shift registers. Each ALM drives all types of interconnects: local, row, column, carry chain, shared arithmetic chain, register chain, and direct link. Figure 2–5 shows a high-level block diagram of the Stratix IV ALM. ***Stratix IV HB v1 – p.2-5***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		One ALM contains two programmable registers. Each register has data, clock, clock enable, synchronous and asynchronous clear, and synchronous load and clear inputs. Global signals, general-purpose I/O pins, or any internal logic can drive the register's clock and clear-control signals. Either general-purpose I/O pins or internal logic can drive the clock enable. For combinational functions, the register is bypassed and the output of the LUT drives directly to the outputs of an ALM. Each ALM has two sets of outputs that drive the local, row, and column routing resources. The LUT, adder, or register outputs can drive these output drivers (refer to Figure 2–6). For each set of output drivers, two ALM outputs can drive column, row, or direct-link routing connections. One of these ALM outputs can also drive local interconnect resources. This allows the LUT or adder to drive one output while the register drives another output. ***Stratix IV HB v1 – p.2-7***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		Building Look-up Tables (LUTs) An overview of how LUTs are built helps describe the key innovations in the ALM. A LUT is typically built out of SRAM bits to hold the configuration memory (CRAM) LUT-mask and a set of multiplexers to select the bit of CRAM that is to drive the output. To implement a k-input LUT (k-LUT)—a LUT that can implement any function of k inputs—2^k SRAM bits and a 2^k:1 multiplexer are needed. Figure 2 shows a 4-LUT, which consists of 16 bits of SRAM and a 16:1 multiplexer implemented as a tree of 2:1 multiplexers. The 4-LUT can implement any function of 4 inputs (A, B, C, D) by setting the appropriate value in the LUT-mask. To simplify the 4-LUT in Figure 2, it can also be built from two 3-LUTs connected by a 2:1 multiplexer. *Figure 2. Building a LUT*  a′b′c′d′ + abcd + abc′d′ = 1000 0000 0000 1001 = 0x8009 Similarly, larger LUTs can be built out of smaller ones, as shown in Figure 3. For example, a 5-LUT can be built with two 4-LUTs and a multiplexer, while a 6-LUT can be built with two 5-LUTs and a multiplexer. Technically, what matters is the total number of CRAM bits in the LUT and that they are used to implement an arbitrary function of six inputs. ***FPGA Architecture WP2006 – p. 3***

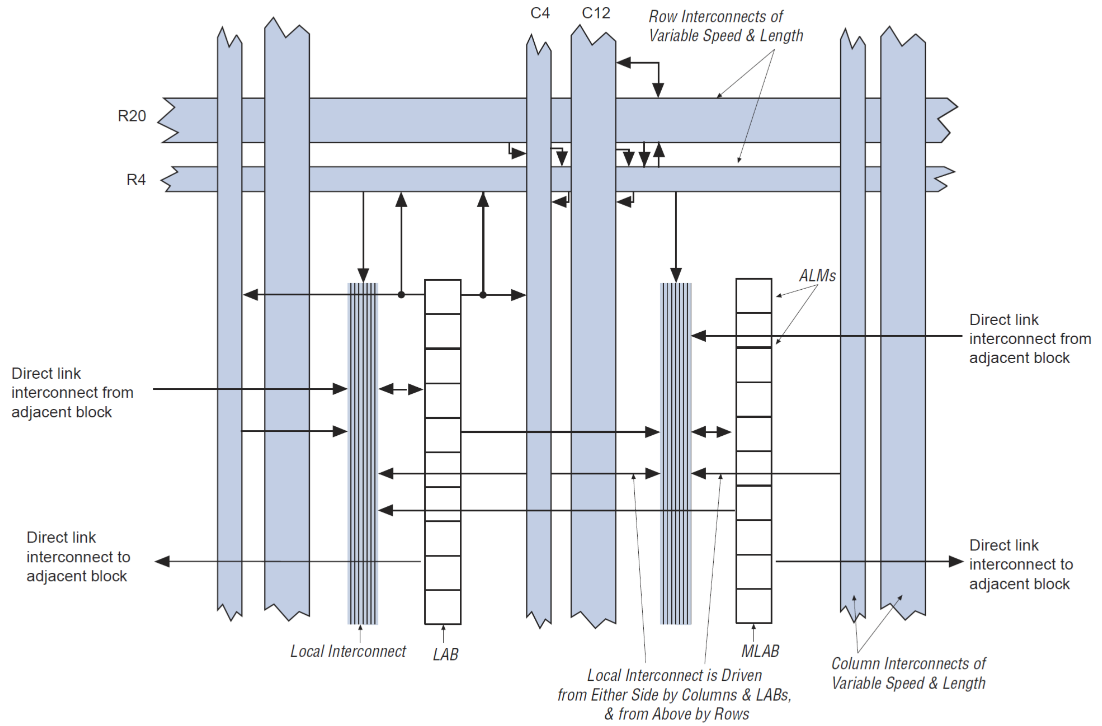
#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		The ALM Advantage The Stratix II ALM has put Altera at least one generation ahead of the competition in FPGA architecture. Introduced more than two years ago, it is significantly more flexible and, as a result, more area efficient than the recently introduced Xilinx Virtex-5 logic element (also called a LUT flipflop pair), which consists of a basic 6-LUT, carry logic, and a single register as shown in Figure 8. In comparison, the combinational logic portion of the ALM has 8 inputs and supports all 6-input functions plus many other combinations of smaller functions using its 2 outputs. The combinational logic portion of the Virtex-5 logic element, a basic 6-LUT, also has 64 bits of CRAM and two outputs like the ALM, but only contains 6 inputs and has a limited ability to implement more than one logic function. One of its outputs is the output of the 6-LUT and the other is the 5-LUT corresponding to the lower half of the configuration RAM. ***FPGA Architecture WP2006 – p. 8***

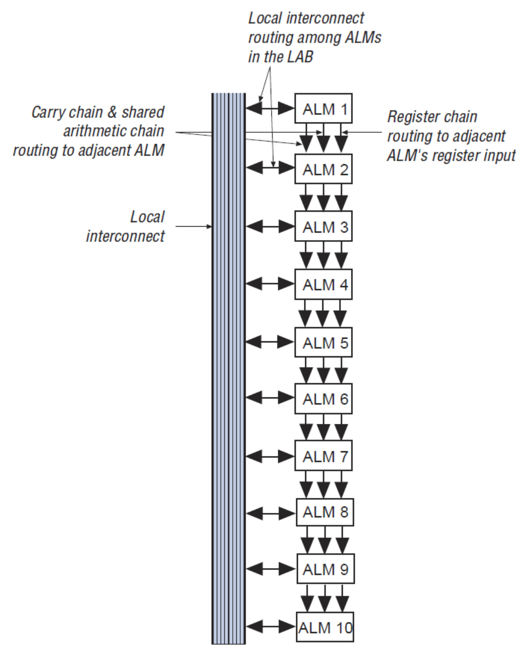
#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		 Figure 9. 6,2 ALM with 2 outputs and SLM ***Hutton2004 – p. 5***

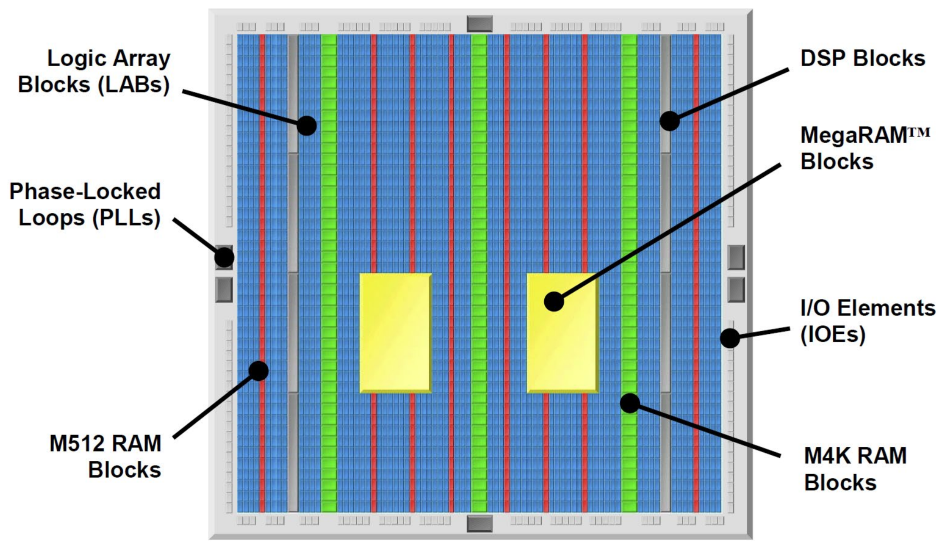
#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
7pre	A **programmable logic circuit device** comprising:	Altera® Stratix® IV FPGAs deliver a breakthrough level of system bandwidth and power efficiency for high-end applications, allowing you to innovate without compromise. Stratix IV FPGAs are based on the Taiwan Semiconductor Manufacturing Company (TSMC) 40-nm process technology and surpass all other high-end FPGAs, with the highest logic density, most transceivers, and lowest power requirements. The Stratix IV device family contains three optimized variants to meet different application requirements: ■ Stratix IV E (Enhanced) FPGAs—up to 813,050 logic elements (LEs), 33,294 kilobits (Kb) RAM, and 1,288 18 x 18 bit multipliers ■ Stratix IV GX transceiver FPGAs—up to 531,200 LEs, 27,376 Kb RAM, 1,288 18 x 18-bit multipliers, and 48 full-duplex clock data recovery (CDR)-based transceivers at up to 8.5 Gbps ■ Stratix IV GT—up to 531,200 LEs, 27,376 Kb RAM, 1,288 18 x 18-bit multipliers, and 48 full-duplex CDR-based transceivers at up to 11.3 Gbps **Architecture Features** The Stratix IV device family features are divided into high-speed transceiver features and FPGA fabric and I/O features. ☞ The high-speed transceiver features apply only to Stratix IV GX and Stratix IV GT devices. ***Stratix IV HB v1 – p.1-1***

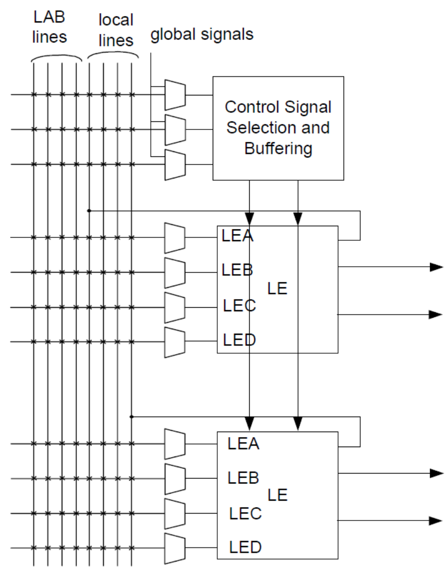
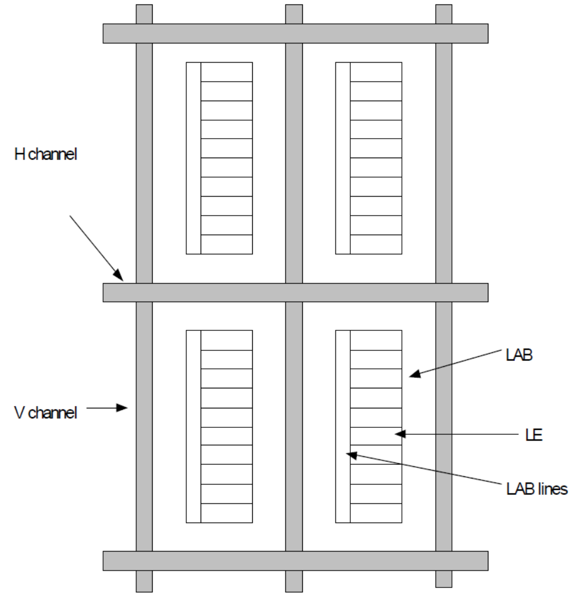
#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
7A	a **plurality of logic blocks**;	**FPGA Fabric and I/O Features** The following sections describe the Stratix IV FPGA fabric and I/O features. **Device Core Features** ■ Up to 531,200 LEs in Stratix IV GX and GT devices and up to 813,050 LEs in Stratix IV E devices, efficiently packed in unique and innovative adaptive logic modules (ALMs) ■ Ten ALMs per logic array block (LAB) deliver faster performance, improved logic utilization, and optimized routing ■ Programmable power technology, including a variety of process, circuit, and architecture optimizations and innovations ■ Programmable power technology available to select power-driven compilation options for reduced static power consumption **Embedded Memory** ■ TriMatrix embedded memory architecture provides three different memory block sizes to efficiently address the needs of diversified FPGA designs: ■ 640-bit MLAB ■ 9-Kb M9K ■ 144-Kb M144K ■ Up to 33,294 Kb of embedded memory operating at up to 600 MHz ■ Each memory block is independently configurable to be a single- or dual-port RAM, FIFO, ROM, or shift register ***Stratix IV HB v1 – p.1-8***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
7B	a plurality of **routing wires** connected to each of said logic blocks;	**Logic Array Blocks** Each LAB consists of ten ALMs, various carry chains, shared arithmetic chains, LAB control signals, local interconnect, and register chain connection lines. The local interconnect transfers signals between ALMs in the same LAB. The direct link interconnect allows the LAB to drive into the local interconnect of its left and right neighbors. Register chain connections transfer the output of the ALM register to the adjacent ALM register in the LAB. The Quartus® II Compiler places associated logic in the LAB or adjacent LABs, allowing the use of local, shared arithmetic chain, and register chain connections for performance and area efficiency. Figure 2–1 shows the Stratix IV LAB structure and the LAB interconnects. ***Stratix IV HB v1 – p.2-1***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		Figure 2–1. Stratix IV LAB Structure  ***Stratix IV HB v1 – p.2-2***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		ALM Interconnects There are three dedicated paths between ALMs: register cascade, carry chain, and shared arithmetic chain. Stratix IV devices include an enhanced interconnect structure in LABs for routing shared arithmetic chains and carry chains for efficient arithmetic functions. The register chain connection allows the register output of one ALM to connect directly to the register input of the next ALM in the LAB for fast shift registers. These ALM-to-ALM connections bypass the local interconnect. The Quartus II compiler automatically takes advantage of these resources to improve utilization and performance. Figure 2–15 shows the shared arithmetic chain, carry chain, and register chain interconnects. **Figure 2–15.** Shared Arithmetic Chain, Carry Chain, and Register Chain Interconnects  ***Stratix IV HB v1 – p.2-18***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
7C	a plurality of switch circuits provided at an intersection of each of said routing wires;	***See claim element 7B.***  **Figure 1 Overview of Stratix Die** ***Lewis2003 – p. 3***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		 2(a) - LAB and Intra-LAB Routing  2(b) - Global Routing Structure Figure 2 – Local and Global Routing Structures ***Lewis2003 – p. 4***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)

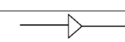
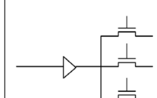
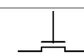
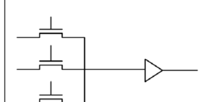
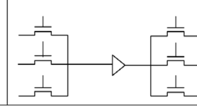
A LAB-based routing architecture consists of a tightly connected block of logic elements connected to a less connected routing fabric. Stratix has two levels of hierarchy of routing resources. The lowest level of the architecture is a logic element (LE) which comprises a single 4-input LUT and flip-flop. The details of the Stratix LE will be described in Section 4. The first level of routing hierarchy is formed by a collection of 10 LEs which are grouped into a logic array block (LAB). These have connectivity between the local routing wires within the LAB. Figure 2(a) shows an overview of a LAB. These routing wires consist of LAB lines which route signals external to the LE to the input pins of the LEs, and local lines, which route the outputs of the LEs to inputs of LEs within the same LAB. The second level is formed by a collection of rows and columns of routing wires to connect the inputs and outputs of the LABs, shown in Figure 2(b). The rows and columns will be referred to as H or V wires (horizontal and vertical) for brevity in this paper.

Lewis2003 – p. 4

3.3 Detailed Routing Architecture

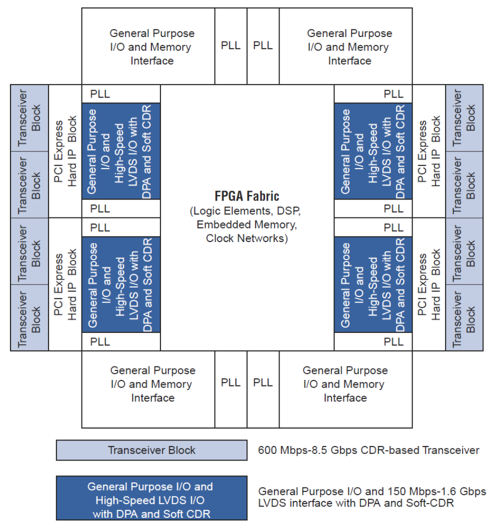
A primary contributor to overall FPGA performance is the choice of routing switches. FMT can model routing switches containing any combination of input muxing pass transistors, buffers, and output demux transistors. There are 8 combinations of these features, of which 5 unique types are electrically feasible, as shown in Table 1. These are (a) buffers (b) buffers and output pass transistors (c) pass transistors (d) direct drive muxes (e) pass transistor muxes with buffers and output demuxes. PMOS pullups (not shown) are used to restore levels after the input multiplexers driving buffers. Buffers do not have any ability to perform programmable routing but may be present in the architecture at various places to repower signals. Thus the prime aspect to evaluate is the choice of the five possible routing switches.

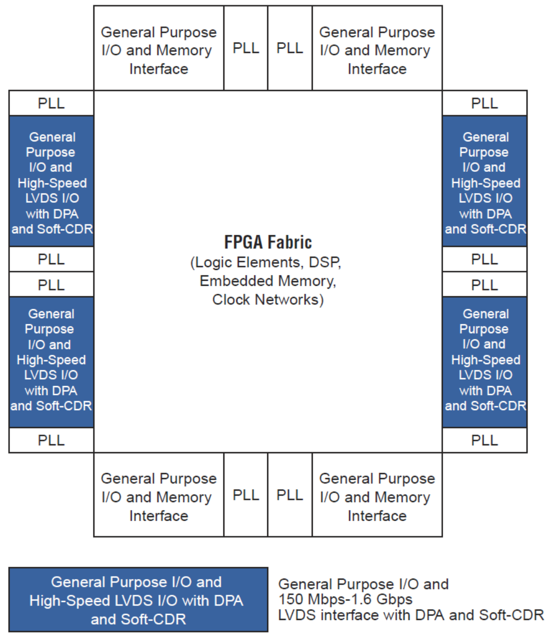
Table 1 - Routing Switch Topologies

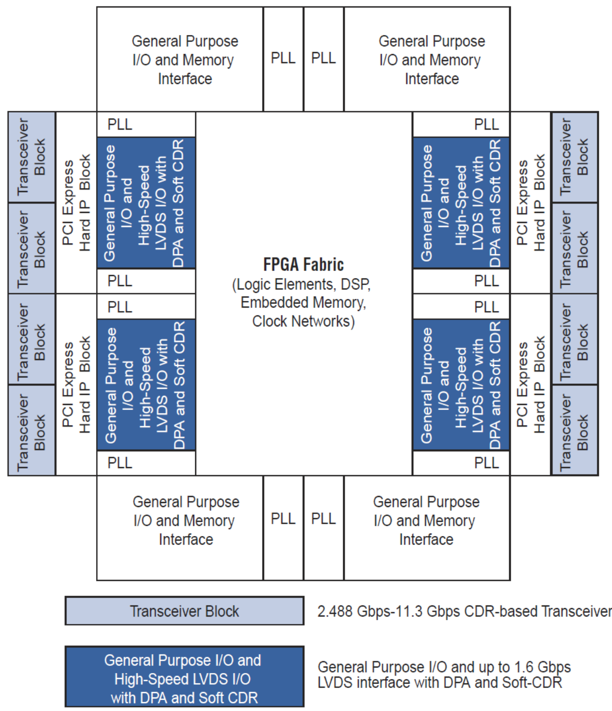
Input Pass Trans?	Buffer?	Output Pass Trans?	Name	Circuit Diagram
N	Y	N	buffer	
N	Y	Y	buffered switch	
Y	N	N	pass transistor	
Y	Y	N	direct drive mux	
Y	Y	Y	mux-demux	

Lewis2003 – p. 6

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		See also similar evidence in Lewis2005.
7D	a plurality of connection blocks provided between an I/O line of each of said logic blocks and each of said routing wires; and	***See elements 7B and 7C.***
7E	an I/O block performing an input/output operation with external equipment,	***See generally Stratix IV HB v1 pp. 6-1 though 6-46 "Chapter 6. I/O Features in Stratix IV Devices"*** **Feature Summary** The following list summarizes the Stratix IV device family features: ■ Up to 48 full-duplex CDR-based transceivers in Stratix IV GX and GT devices supporting data rates up to 8.5 Gbps and 11.3 Gbps, respectively ■ Dedicated circuitry to support physical layer functionality for popular serial protocols, such as PCI-Express (PIPE) Gen1 and Gen2, Gigabit Ethernet, Serial RapidIO, SONET/SDH, XAUI/HiGig, (OIF) CEI-6G, SD/HD/3G-SDI, Fibre Channel, SFI-5, and Interlaken ■ Complete PCI Express (PIPE) protocol solution with embedded PCI Express hard IP blocks that implement PHY-MAC layer, Data Link layer, and Transaction layer functionality For more information, refer to the *PCI Express Compiler User Guide*. ***Stratix IV HB v1 - p. 1-1***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		Figure 1–1. Stratix IV GX Chip View *(Note 1)*  **Note to Figure 1–1:** (1) Resource counts vary with device selection, package selection, or both. ***Stratix IV HB v1 - p. 1-3***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		Figure 1–2. Stratix IV E Chip View *(Note 1)*  **Note to Figure 1–2:** (1) Resource counts vary with device selection, package selection, or both. ***Stratix IV HB v1 - p. 1-4***

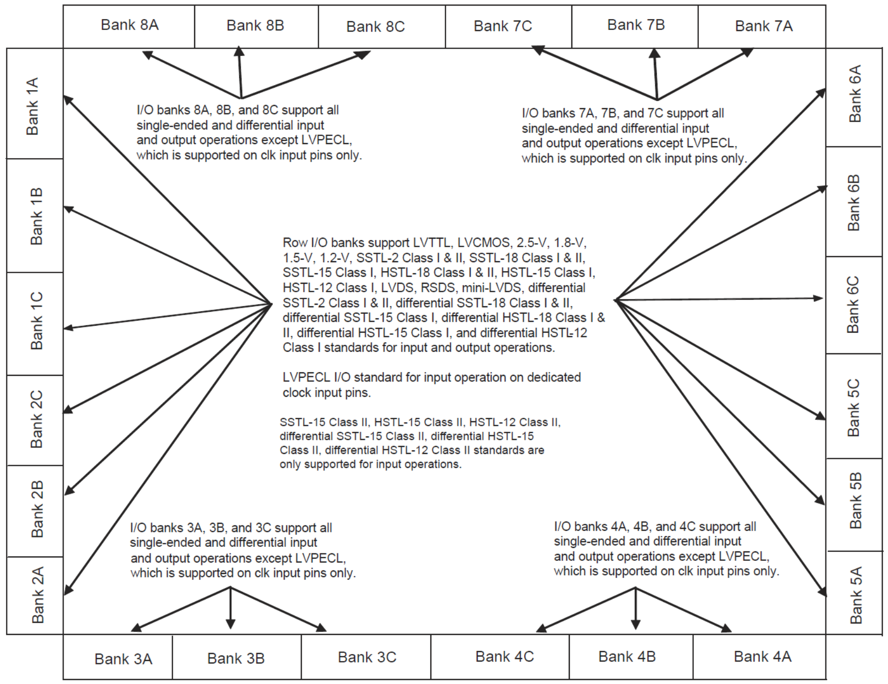
#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		Figure 1–3. Stratix IV GT Chip View *(Note 1)*  **Note to Figure 1–3:** (1) Resource counts vary with device selection, package selection, or both. ***Stratix IV HB v1 - p. 1-5*** ***See also Stratix IV HB v1 - pp. 1-6 though 1-18.***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		I/O Features ■ Sixteen to 24 modular I/O banks per device with 24 to 48 I/Os per bank designed and packaged for optimal simultaneous switching noise (SSN) performance and migration capability ■ Support for a wide range of industry I/O standards, including single-ended (LVTTL/CMOS/PCI/PCIX), differential (LVDS/mini-LVDS/RSDS), voltage-referenced single-ended and differential (SSTL/HSTL Class I/II) I/O standards ■ On-chip series (R_S) and on-chip parallel (R_T) termination with auto-calibration for single-ended I/Os and on-chip differential (R_D) termination for differential I/Os ■ Programmable output drive strength, slew rate control, bus hold, and weak pull-up capability for single-ended I/Os ■ User I/O:GND:V_{CC} ratio of 8:1:1 to reduce loop inductance in the package—PCB interface ■ Programmable transmitter differential output voltage (V_{OD}) and pre-emphasis for high-speed LVDS I/O **High-Speed Differential I/O with DPA and Soft-CDR** ■ Dedicated circuitry on the left and right sides of the device to support differential links at data rates from 150 Mbps to 1.6 Gbps ■ Up to 98 differential SERDES in Stratix IV GX devices, up to 132 differential SERDES in Stratix IV E devices, and up to 47 differential SERDES in Stratix IV GT devices ■ DPA circuitry at the receiver automatically compensates for channel-to-channel and channel-to-clock skew in source synchronous interfaces ■ Soft-CDR circuitry at the receiver allows implementation of asynchronous serial interfaces with embedded clocks at up to 1.6 Gbps data rate (SGMII and Gigabit Ethernet) ***Stratix IV HB v1 - p. 1-9***

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
		■ Programmable transmitter pre-emphasis and receiver equalization circuitry to compensate for frequency-dependent losses in the physical medium ■ Typical physical medium attachment (PMA) power consumption of 100 mW at 3.125 Gbps and 135 mW at 6.375 Gbps per channel ■ 72,600 to 813,050 equivalent LEs per device ■ 7,370 to 33,294 Kbits of enhanced TriMatrix memory consisting of three RAM block sizes to implement true dual-port memory and FIFO buffers ■ High-speed DSP blocks configurable as 9 × 9-bit, 12 × 12-bit, 18 × 18-bit, and 36 × 36-bit full-precision multipliers at up to 600 MHz ■ Up to 16 global clocks (GCLK), 88 regional clocks (RCLK), and 132 periphery clocks (PCLK) per device ■ Programmable power technology that minimizes power while maximizing device performance ■ Up to 1,120 user I/O pins arranged in 24 modular I/O banks that support a wide range of single-ended and differential I/O standards ■ Support for high-speed external memory interfaces including DDR, DDR2, DDR3 SDRAM, RLDRAM II, QDR II, and QDR II+ SRAM on up to 24 modular I/O banks ■ High-speed LVDS I/O support with serializer/deserializer (SERDES), dynamic phase alignment (DPA), and soft-CDR circuitry at data rates up to 1.6 Gbps ■ Support for source-synchronous bus standards, including SGMII, Gigabit Ethernet, SPI-4 Phase 2 (POS-PHY Level 4), SFI-4.1, XSBI, UTOPIA IV, NPSI, and CSIX-L1 ■ Pinouts for Stratix IV E devices designed to allow migration of designs from Stratix III to Stratix IV E with minimal PCB impact ***Stratix IV HB v1 - p. 1-2***

<table>
<tr><th>#</th><th>Claim</th><th>INTEL FPGAs – Adaptive Logic Modules (ALMs)</th></tr>
<tr><td></td><td></td><td>

Figure 6–1. Stratix IV E Devices I/O Banks *(Note 1), (2), (3), (4), (5), (6), (7), (8)*



Notes to Figure 6–1:

(1) Differential HSTL and SSTL outputs are not true differential outputs. They use two single-ended outputs with the second output programmed as inverted.

(2) Column I/O differential HSTL and SSTL inputs use LVDS differential input buffers without differential OCT support.

(3) Column I/O supports LVDS outputs using single-ended buffers and external resistor networks.

(4) Column I/O supports PCI/PCI-X with on-chip clamp diode. Row I/O supports PCI/PCI-X with external clamp diode.

(5) Clock inputs on column I/O are powered by $V_{CCCLKIN}$ when configured as differential clock inputs. They are powered by V_{CCIO} when configured as single-ended clock inputs. All outputs use the corresponding bank V_{CCIO}.

(6) Row I/O supports the true LVDS output buffer.

(7) Column and row I/O banks support LVPECL standards for input clock operation.

(8) Figure 6–1 is a top view of the silicon die that corresponds to a reverse view for flip chip packages. It is a graphical representation only.

Stratix IV HB v1 - p. 6-6 (see also through p. 6-17)

</td></tr>
</table>

Professor Masahiro Iida v. Intel Corporation – Civil Action No. 6:22-cv-00662 (E.D. Tex.)

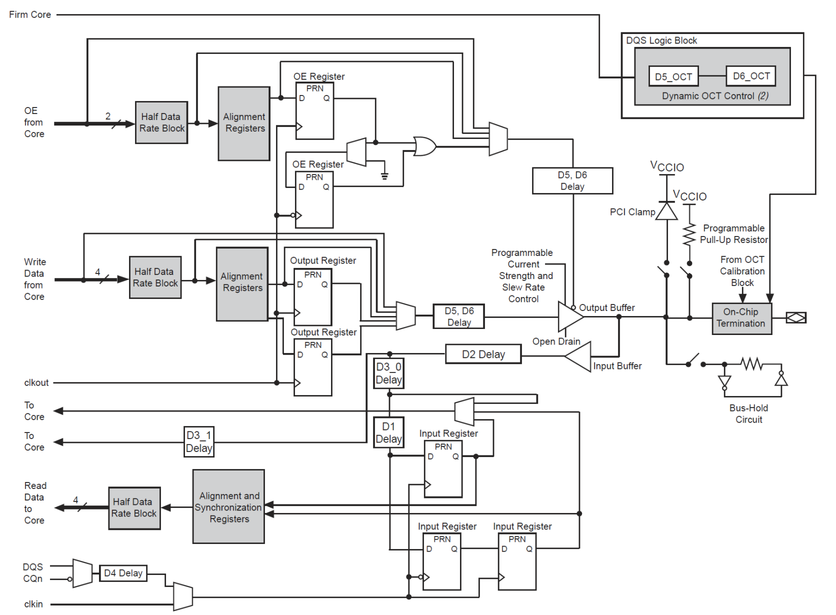
#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)

I/O Structure

The I/O element (IOE) in Stratix IV devices contain a bidirectional I/O buffer and I/O registers to support a complete embedded bidirectional single data rate or DDR transfer. The IOEs are located in I/O blocks around the periphery of the Stratix IV device. There are up to four IOEs per row I/O block and four IOEs per column I/O block. The row IOEs drive row, column, or direct link interconnects. The column IOEs drive column interconnects.

Stratix IV HB v1 - p. 6-17

Figure 6–17. Stratix IV IOE Structure *(Note 1), (2)*



Notes to Figure 6–17:

(1) The D3_0 and D3_1 delays have the same available settings in the Quartus II software.

(2) One dynamic OCT control is available per DQ/DQS group.

Stratix IV HB v1 - p. 6-18

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
7F	wherein each of said logic blocks has a look up table of M inputs and N outputs, comprising:	**See claim element 1pre.**
7G	a plurality of **LUT units**; and	**See claim element 1A.**
7H	an **internal configuration control circuit** controlling an internal **configuration of said plurality of LUT units**, wherein said internal configuration control circuit comprises	**See claim element 1B.**
7I	a **plurality of selectors selecting I/O signals** of said	**See claim element 1C.**

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
	plurality of LUT units, and	
7J	a **selector control circuit** having a **memory**, controlling said plurality of selectors in accordance with **data stored in said memory**, and **defining the internal configuration** of said plurality of LUT units.	**See claim element 1D.**

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
8A	The programmable logic circuit device as claimed in claim 7, wherein said plurality of	**See claim element 2A.**

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
	selectors include: an **input signal selector** provided at **an input of at least one of said LUT units** to select an input signal; and an **output signal selector** provided at an **output of said LUT units** selecting an output signal,	
8B	said input signal selector and said output signal selector being **controlled in accordance with the data stored in said memory**.	**See claim element 2B.**

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
13pre	A method of configuring a **look up table of M inputs and N outputs**, comprising:	**See claim element 1pre.**

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
13A	providing a **plurality of LUT units**; and	**See claim element 1A.**

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
13B	**selectively controlling I/O signals** of said plurality of LUT units to **set a predetermined mode of an internal configuration**.	**See claim element 1B and 1C.**

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
14	The method of configuring a look up table as claimed in claim 13, wherein the **I/O signals of said plurality of LUT units** are **selectively controlled in accordance with data stored in the corresponding look up table**.	**See claim element 2A and 2B.**

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
15	15. The method of configuring a look up table as claimed in claim 13, wherein **an input signal input to at least one of said LUT** units and	**See claim 14.**

#	Claim	INTEL FPGAs – Adaptive Logic Modules (ALMs)
	an output signal output from said LUT unit[s] are **selectively controlled in accordance with data stored in the corresponding look up table**.	